Vision Assisted Navigation for Miniature Unmanned Aerial Vehicles (MAVs)

Final Report (Feb 2007 - Nov 2009)
FA9550-07-1-0167

Clark N. Taylor

***Abstract*—One of the primary difficulties in flying airplanes of all sizes is navigation, or ensuring that the location and attitude of the airplane is known at all times. For large aerial systems, high-quality IMUs have been successfully used for decades. However, the weight, power, cost, and size restrictions of miniature-UAVs (MAVs) preclude the use of high-quality IMUs for navigation. For the past two and a half years, this contract has enabled the investigation of techniques for enabling high-quality navigation using a combination of low-quality inertial sensors, visual sensors, and (when available) GPS. Specific areas of contribution from this project include: (1) real-time techniques for improving pose estimates on MAVs with GPS, (2) two proposed approaches for enabling GPS-denied navigation, (3) quantitative and analytical analyses of different navigation algorithms, and (4) GPS-denied navigation utilizing multiple agents. The contributions in each of these areas is discussed in this report.**

I. Introduction

Recently, Unmanned Aerial Vehicles (UAVs) have seen a dramatic increase in their utilization for military applications. In addition, UAVs are being investigated for multiple civilian applications, including rural search and rescue, forest fire monitoring and agricultural information gathering. Due to their small size, Miniature UAVs (MAVs) are an attractive platform for executing many civilian and military missions. Some of the primary advantages of MAVs include: (1) they are significantly less expensive to purchase than the large UAVs typically used by the military; (2) their small size simplifies transport, launch and retrieval; and (3) they are less expensive to operate than large UAVs.

Accurate pose (location and attitude) estimation is essential for many MAV missions. For example, if the MAV is conducting a surveillance mission and the GPS location of an object in the video is desired, an attitude estimate error of only a few degrees for an MAV flying 500 meters above the ground can lead to dozens of meters of error in the object location estimate. However, the size and weight constraints of an MAV dictate the use of lightweight and therefore inaccurate sensors. To compensate for the inaccurate sensors, prior pose estimation methods have focused largely on fusion of measurements from GPS and inertial measurement units (IMUs)[1], [2], [3], [4].

These approaches, however, suffer from two primary weaknesses. First, many envisioned scenarios for MAVs include operation in tightly constrained spaces such as urban terrain, dense foliage, and inside buildings. In these environments, the signals from GPS satellites will typically be occluded, precluding the use of GPS-based navigation systems. Second, even when GPS is available, the navigation state accuracy of current techniques is very low due to the quality of sensors used to estimate the MAV pose.

To overcome these weaknesses, our research has focused on enabling navigation of MAVs through the fusion of visual and inertial sensors. Visual sensors, such as cameras, possess a number of advantages. The senors themselves are lightweight and low power. Many existing MAV systems carry an on-board camera for visual observations, which can be used simultaneously for pose estimation. In addition, the source of errors in the inertial and visual sensors are generally uncorrelated, enabling fusion of the sensors together for more accurate pose estimation.

This report contain a description of several different areas of research into visual and inertial sensor fusion. In Section II, we discuss a method that can be utilized in real-time to achieve more accurate pose estimates when GPS is available. In Section III, we present a preliminary study demonstrating some potential of visual/inertial sensor fusion for improved navigation without GPS. In Section IV, we describe several modifications made to prior visual/inertial fusion navigation techniques to enable fusion on-board an MAV. Sections V and VI present a simulation-based and analytical comparison, respectively, of our improved technique with SLAM-based navigation techniques. Finally, Sections VII and VIII introduce navigation techniques for multiple vehicles, enabling reduced and bounded drift navigation techniques, respectively. We conclude this report by listing all publications resulting from this research and discussing future work in Sections IX and X.

II. Real-time navigation improvement for MAVs with GPS

Much of the prior research in using visual information for pose estimation has been in visual odometry (VO) [5], [6], where a pose is computed directly from the motion of points in the camera view. This makes it possible to use the camera as a standalone sensor. However, all visual odometry methods share several key weaknesses. First, VO systems can only compute motion relative to the pose of the camera from the previous frame. This means that the system depends on accurate initialization of the camera pose at the first frame. It also means that the current pose is computed by integrating all previous poses, which causes error in the poses to grow without bound. The second limitation is that while camera rotation can be unambiguously reconstructed, only

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 15-07-2010	**2. REPORT TYPE** Final Report	**3. DATES COVERED** *(From - To)* Feb 2007- Nov 2009

4. TITLE AND SUBTITLE Vision Assisted Navigation for Miniature Aerial Vehicles	**5a. CONTRACT NUMBER** FA9550-07-1-0167
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Clark N. Taylor, PhD	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Brigham Young University A-285 ASB Provo, UT 84602	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFOSR	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Distribution A: Approved for Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT

One of the primary difficulties in flying airplanes of all sizes is navigation, or ensuring that the location and attitude of the airplane is known at all times. For large aerial systems, high-quality IMUs have been successfully used for decades. However, the weight, power, cost, and size restrictions of miniature-UAVs (MAVs) preclude the use of high-quality IMUs for navigation. For the past two and a half years, this contract has enabled the investigation of techniques for enabling high-quality navigation using a combination of low-quality inertial sensors, visual sensors, and (when available) GPS. Specific areas of contribution from this project include: (1) real-time techniques for improving pose estimates on MAVs with GPS, (2) two proposed approaches for enabling GPS-denied navigation, (3) quantitative and analytical analyses of different navigation algorithms, and (4) GPS-denied navigation utilizing multiple agents. The contributions in each of these areas is discussed in this report.

15. SUBJECT TERMS

GPS-denied navigation; inertial and visual sensor fusion; estimation theory

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Clark Taylor
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	None	53	**19b. TELEPHONE NUMBER** *(include area code)* 801-422-3903

the direction of the camera's translation, not the magnitude, can be reconstructed without additional information.

To avoid the problems inherent in VO, we focus on fusing together information from the camera with a GPS/IMU system to achieve more accurate pose estimates. Prior methods dealing with fusion of visual information and GPS/IMU estimates have focused on two approaches. The first uses visual information to gauge the accuracy of the GPS/IMU estimates [7], [8], [9]. It requires that the 3D location of points in the camera view be accurately estimated, something which can be difficult to do in MAV video. (Note that most prior methods assume either a stereo camera system or a wide-area field of view, enabling accurate estimation of 3-D locations. Neither of these assumptions are typically valid for MAVs.) The second approach [10] directly estimates a pose by aligning successive images. This method can be quite accurate, but the iterative methods employed to compute a pose estimate require extensive computation time. The processing units found in an MAV are not capable of doing such computation fast enough to compute pose estimates in real time.

We propose a method to combine GPS/IMU pose estimates with visual information that does not require computation of 3D point locations or direct image registration. Instead, optical flow measurements are combined to create a homography matrix which can be input directly into an Unscented Kalman Filter [11] (UKF) as the measurement variable. Treating pose estimates from the GPS/IMU systems as a state variable, the UKF is able to combine the information from both of these sensors to increase the accuracy of pose estimation.

An additional advantage of our method is that it does not require decomposition of the homography matrix into rotation and translation components. While several previous methods have used decomposition of the homography matrix as a way to estimate pose from vision [12], [13], [14], [15], [16], homography decomposition is a poorly-conditioned problem when the magnitude of the camera's translation is small compared to the distance of the camera from the scene being imaged. Because of the altitudes at which MAVs fly and their relatively slow speeds, this ill-conditioned case is typical in MAV video.

The remainder of this section is outlined as follows. In section II-A we describe our overall system for improving pose estimates using the UKF. Section II-B describes a novel method to transform the covariance of feature tracking measurements into covariance of the corresponding homography. We present some results achieved using our fusion system in Section II-C.

A. Vision/GPS/IMU Fusion System

To improve the accuracy of MAV pose estimation, we propose to fuse motion information captured by a camera with GPS/IMU estimates of pose as shown in Figure 1. At the core of our system is an Unscented Kalman Filter, chosen for its ability to combine data with varying uncertainty in highly non-linear systems. The *state* of the UKF is a 12x1 vector representing two most recent poses of the MAV. This state is initialized as:

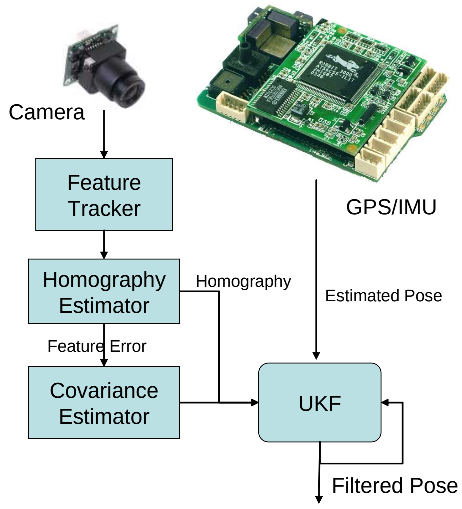


Fig. 1. Layout of the vision/GPS/IMU fusion system.

$$\mathbf{x}_0 = \left[\mathbf{y}_0, \mathbf{y}_1\right]^T , \tag{1}$$

where $\mathbf{y}_t$ represents the pose of the MAV at time t. A pose includes 3 parameters for the location of the MAV $(\hat{x}, \hat{y}, \hat{z})$ and 3 parameters for the attitude (ϕ, θ, ψ). It is necessary to keep two pose estimates in the state due to the fact that two video frames are required in order to extract information about camera motion. After applying the measurement update step of the UKF, the state of the UKF is two *refined* pose estimates, defined as

$$\mathbf{x'}_0 = \left[\mathbf{y}'_0, \mathbf{y}'_1\right]^T . \tag{2}$$

Whenever a new pose estimate is received from the GPS/IMU system, the state of the UKF is updated to contain the most recent pose measurement, and the previous *refined estimate*

$$\mathbf{x}_t = \left[\mathbf{y}'_t, \mathbf{y}_{t+1}\right]^T . \tag{3}$$

By consistently keeping a refined pose estimate in the state of the UKF, the pose estimates become more accurate over time. Corresponding steps are taken with the covariance matrix associated with the state of the UKF.

To generate the refined pose estimates introduced above, it is necessary to (1) define the state-measurement function which transforms a state vector to a measurement vector (2) implement the *measurement update* step of the UKF. These are covered in the next two sections.

1) State-measurement function: Each frame in a video sequence does not explicitly contain information about the pose of the camera. However, by comparing two frames of video, a *homography matrix* that encapsulates information about the relative movement of the camera can be computed. A homography matrix maps the location of all points in an image to their location in the second image, assuming that the objects being observed by the camera are co-planar. Due to the flight characteristics of MAVs co-planarity is usually a valid assumption. The homography relationship is represented by the equation

$$\Upsilon' = H\Upsilon, \tag{4}$$

where $\Upsilon$and Υ' are the location of points in images 1 and 2, respectively, and H is the homography matrix. (We discuss how the homography matrix is estimated in section II-B.)

In addition to mapping pixel locations between images, a homography is related to the displacement between cameras by the formula

$$H = K_2\left(R + \frac{1}{d}TN^T\right)K_1^{-1} \tag{5}$$

where R is the rotation from camera 1 to camera 2, T is the translation from camera 1 to camera 2 in camera 2's coordinate system, N is the unit normal of the plane being imaged in camera 1's coordinate system, and d is the distance from the optical center of camera 1 to the plane. K_1 and K_2 represent the calibration matrix of each camera and are assumed to be known *a priori*. In our case, the same camera captured both images so $K_1 = K_2 = K$. In order to use the poses from the GPS/IMU, this formulation must be modified due to the fact that the pose parameters are relative to world coordinates.

For a state vector $\mathbf{x}_t = [\mathbf{y}_t, \mathbf{y}_{t+1}]$ in the UKF, two rotation matrices $\left(R_{wc_t}, R_{wc_{t+1}}\right)$ can be derived that rotate from the world frame to the coordinate frame of the camera at time t and $t+1$. Similarly, two translation vectors $\left(T_{wc_t}, T_{wc_{t+1}}\right)$ for the cameras, in world coordinates, can be derived. Any point in 3-space that is observed by both cameras can be rotated from one camera's coordinate system to the other by a rotation matrix R, which can be expressed in terms of (R_{wc_1}, R_{wc_2}) as

$$R = R_{wc_2}R_{wc_1}^T. \tag{6}$$

The vector T in Equation (5) can be expressed in terms of (T_{wc_1}, T_{wc_2}) as

$$T = R_{wc_2}\left(T_{wc_1} - T_{wc_2}\right). \tag{7}$$

In order to obtain N, the (known) normal of the plane in world coordinates N_w must be rotated into camera 1 coordinates.

$$N = R_{wc_1}N_w \tag{8}$$

Because we assume that N_w is $[0, 0, 1]$ (i.e. the ground is not sloped), the parameter d is simply the z-component of camera 1's translation in world coordinates. Substituting Equations (6-8) into Equation (5) and simplifying yields

$$H = KR_{wc_2}\left(I - \frac{1}{d}\left(T_{wc_1}^w - T_{wc_2}^w\right)N_w^T\right)R_{wc_1}^T K^{-1}. \tag{9}$$

There is a scale ambiguity inherent in the homography [17], so to eliminate it we normalize the homography so that the ninth element of the matrix is equal to one.

$$H = H/h_{3,3}. \tag{10}$$

The first 8 elements of the homography matrix are then taken as the predicted measurement vector.

2) Measurement update: The measurement update step of the Kalman Filter uses the current state, the current homography measurement and their associated covariance matrices to compute a more accurate estimate of the current state. In order to do this it is necessary to transform the state-covariance matrix into the measurement space, something which is difficult to do in a non-linear system. The UKF achieves this transformation by sampling the distributions of the state variables to come up with a number of sample states. Using the state-measurement function, these sample states are then transformed into sample measurements. The mean of the sample measurements becomes the predicted measurement, and the average covariance between the mean and the sample measurements form the predicted measurement covariance.

The process by which this is done is as follows:

1) Create the augmented state vector $\mathbf{x}^a$ by combining the current state $\mathbf{x}_t$ and the mean of the measurement noise ν

$$\mathbf{x}^a = \left[\mathbf{y}'_{t-1}, \mathbf{y}_t, \nu\right]^T. \tag{11}$$

2) Create an augmented covariance matrix from the covariance of the filtered pose P_{t-1}, the current pose P_t and the homography P_h

$$P^a = \begin{bmatrix} P_{t-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & P_t & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & P_h \end{bmatrix}. \tag{12}$$

3) Construct the set of sample points χ_i

$$\begin{aligned} \chi_0 &= \mathbf{x}^a, \\ \chi_i &= \mathbf{x}^a + \left(\sqrt{(L+\kappa)\,P^a}\right)_i, \\ \chi_{i+L} &= \mathbf{x}^a - \left(\sqrt{(L+\kappa)\,P^a}\right)_i, \end{aligned} \tag{13}$$

and a set of weights

$$\begin{aligned} W_0 &= \kappa/\left(L+\kappa\right), \\ W_i &= 1/2\left(L+\kappa\right), \text{ and} \\ W_{i+L} &= 1/2\left(L+\kappa\right) \end{aligned} \tag{14}$$

where $\left(\sqrt{(L+\kappa)\,P^a}\right)_i$ is the i^{th} column of the matrix square root of $(L+\kappa)\,P^a$ (any matrix square root can be used). L is the number of elements in the augmented state vector and κ is set such that $L + \kappa = 3$ as suggested by Julier and Uhlman [11].

4) Transform the sample points with the state-measurement transformation

$$\gamma_i = \mathcal{F}\left(\chi_i\right), \tag{15}$$

where $\mathcal{F}(\chi_i)$ represents the formulation of a homography from two poses (Eqn. (9)), followed by scale normalization (Eqn. (10)).

5) Find the mean of the sample homographies

$$\hat{\mathbf{h}} = \sum_{i=0}^{2L} W_i \gamma_i. \tag{16}$$

6) Compute the covariance from the weighted outer product of the transformed points

$$P_{hh} = \sum_{i=0}^{2L} W_i \left(\gamma_i - \hat{\mathbf{h}}\right)\left(\gamma_i - \hat{\mathbf{h}}\right)^T. \tag{17}$$

7) Compute the state-measurement cross-correlation from the weighted outer product of both sets of points

$$P_{xh} = \sum_{i=0}^{2L} W_i \left(\chi_i - \mathbf{x}\right)\left(\gamma_i - \hat{\mathbf{h}}\right)^T. \tag{18}$$

8) Compute the optimal Kalman gain

$$K = P_{xh} P_{hh}^{-1}. \tag{19}$$

9) Update the pose estimates

$$\hat{\mathbf{x}} = \mathbf{x} + K\left(\mathbf{h} - \hat{\mathbf{h}}\right). \tag{20}$$

10) Update the state covariance

$$P_{xx} = P_{xx} - K P_{hh} K^T. \tag{21}$$

The elements of the refined state vector corresponding to the current pose and the associated covariances are then fed back into the filter for the next iteration. A result of this feedback system is that each pose will be filtered twice. Therefore, for increased accuracy, a filtered pose can be used after its second time through the filter.

In order for the filter to work properly it is essential to have an accurate estimate of the measurement covariance. This allows an appropriate level of confidence to be placed in the state correction caused by the measurement. Section II-B discusses a novel method we have developed to estimate homography covariance from the outputs of our RANSAC system.

B. Transformation of Uncertainty from Feature tracking to Homography

To utilize the homography matrix in the UKF, two items must be computed from the visual information: (1) an estimate of the homography matrix and (2) an estimate of the covariance of the homography matrix. These two blocks are illustrated by the *homography estimator* and *covariance estimator* blocks in Figure 1.

To compute the estimated homography, features identified by the Harris Corner Detector [18] are tracked across images using a pyramidal implementation of the Lucas-Kanade optical flow algorithm [19]. Because video frames and pose estimates from the GPS/IMU unit are received at different times, the Harris corner detector is run whenever a new pose estimate is received, and the selected set of features is tracked across every frame until the next pose estimate is received. Once the second pose estimate is received, a RANSAC [20] algorithm is used to reject outliers and the homography that best fits the remaining feature correspondences is computed using the *normalized DLT algorithm* outlined in [17].[1] By setting a minimum number of inliers, the RANSAC algorithm can also detect most mis-registrations. In such cases, only information from the GPS/IMU is used to update the filter.

Once an estimate of the homography has been computed, the covariance associated with that estimate must be determined. To do this we propose using information obtained from the RANSAC and DLT algorithms to determine the variance of the feature tracking process and then use techniques from the UKF to transform this into a covariance matrix for the estimated homography.

From the homography estimation system we obtain a list of inlier features. For each inlier a residual error term can be computed by finding the distance between the tracked feature location and the feature location as predicted by the homography. By computing the standard deviation of this residual error (σ), we are able to characterize the noise present in feature locations. The standard deviation is computed as:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} \left\|\Upsilon_i' - H\Upsilon_i\right\|^2}{n}}, \tag{22}$$

where n is the number of inlier features. From this standard deviation, we compute a covariance matrix representing the uncertainty in all feature locations as

$$P_{\Upsilon} = \frac{\sigma^2}{2} I. \tag{23}$$

This covariance matrix is of size $2n \times 2n$, representing the covariance in both the x and y locations in the image. Similarly, σ^2 is divided by two to evenly distribute the uncertainty in the $\hat{x}$ and $\hat{y}$ directions.

Due to the non-linearity of the transformation from point correspondences to a homography matrix, it is difficult to directly transform the covariance of feature locations into the homography space. Instead of a direct transformation, we propose to use the sampling technique of the UKF. We create a set of sample point correspondences, transforming them into homography matrices and computing the covariance of the resulting matrices.

To perform the covariance transformation, a set of $4n+1$ sample points χ_i are created as

$$\begin{gathered} \chi_0 = \boldsymbol{\Upsilon}', \\ \chi_i = \boldsymbol{\Upsilon}' + \left(\sqrt{(L+\kappa)P}\right)_i, \text{ and} \\ \chi_{i+n} = \boldsymbol{\Upsilon}' - \left(\sqrt{(L+\kappa)P}\right)_i \end{gathered} \tag{24}$$

[1]Harris corner detection, pyramidal Lucas-Kanade feature tracking, and the normalized DLT algorithm are well known in the computer vision field and are not covered in more detail here. We encourage readers to visit the citations for more information on these techniques.

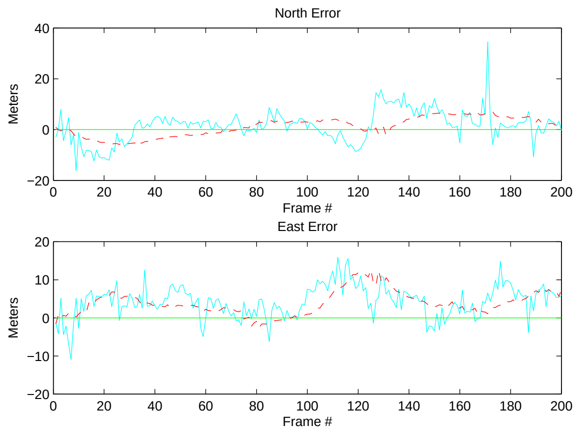


Fig. 3. Evaluation of filter performance using MAV flight data. The MAV was commanded to orbit a target of known location. The error in target geolocation is shown for 200 frames with filtering (dashed line) and without (solid line).

where $L = 2n$ and $\kappa = 3 - L$ as in Equation (13), and $\boldsymbol{\Upsilon}' = [\Upsilon'_1 \dots \Upsilon'_n]$.

A homography is computed for each χ_i by combining the sample point with with the feature points $[\Upsilon_1 \dots \Upsilon_n]$ using the normalized DLT algorithm[17], creating $4n{+}1$ vectorized homography matrices (γ_i). Each homography is normalized using Equation (10) and the covariance of the homographies is then computed as

$$C = \frac{1}{2n} \sum_{i=1}^{2n} (\gamma_i - \gamma_0)(\gamma_i - \gamma_0)^T . \tag{25}$$

C. Results

We evaluated the performance of the filtering framework in two ways. The first method used simulated data to evaluate the performance of the filter independent from feature tracking. A series of true poses for an MAV were generated and from these poses a true homography was calculated for each sequential frame pair. For each true homography a set of point correspondences was generated. The true poses and the point correspondences were then corrupted with Gaussian noise and used as inputs to the filter. Results of the simulation are shown in Figure 2. The noise added to the true pose values had a standard deviation of 1 meter in location and 2 degrees in attitude, while the noise in the point correspondences had a standard deviation of 0.5 pixels. In order to obtain a good statistical sampling, 5,000 iterations of the simulation were run. The results show that the filter was able to significantly reduce the noise in the pose estimates. The standard deviation of the error in filtered poses was reduced to approximately 0.5 meters in location and 0.66 degrees in attitude.

The second method used real MAV flight data. It was not possible to precisely determine the true pose of the MAV, so the performance of the filter was determined by using the estimated poses to solve a geolocation problem. An MAV flying at an altitude of 70 meters was used to collect 640×480 resolution video of a target with a known location. GPS/IMU pose estimates of the MAV were generated by a Kestrel autopilot [3] and transmitted to a ground station at the rate of 4 Hz. The Kestrel uses MEMS based accelerometers and rate gyros as well as differential pressure sensors and consumer grade GPS to compute pose estimates. These estimates were synchronized with the video frames to within approximately 75 milliseconds. About 200 frames with pose estimates were captured.

Using the method in [21] a location for the target was computed using the filtered pose estimate, which was then compared to the target's actual location. For this test, the data was post-processed in the UKF framework using Matlab on a computer with a 3.2 GHz Pentium 4 processor. Our test system processed the data at 2 frames-per-second, although a careful C-code implementation should be able to achieve significantly higher throughput. The results of this test in Figure 3, show that the filtered poses reduce the error in target geolocation as well as significantly reducing the short-term variance in the estimates based on the GPS/IMU alone. Due to the nature of the autopilot sensors, there remains a slowly varying bias term in the error which cannot be estimated by this formulation of the filter. Nevertheless, we are still able to reduce the average absolute error in target geolocation by 29.4%.

III. The Advantages of IMU/Visual Sensor Fusion over Vision-Only Based Navigation

While the prior section described a method for improving MAV pose estimates when GPS is available, we were also interested in demonstrating the potential visual and inertial sensor fusion for GPS-denied scenarios. In this section, we demonstrate that fusing inertial and visual sensors leads to improved navigation accuracy.

Many of the advantages associated with fixed-wing MAVs stem from their ability to operate autonomously at two levels. On a basic level, MAVs must be able to autonomously perform low-level flight tasks such as taking off, flying straight & level, climbing, descending, banking, etc. On a higher level, MAVs must be able to combine these maneuvers in order to fly to specific locations, follow specific trajectories, and otherwise *navigate* autonomously. In order to perform these tasks, MAVs must be able to estimate their own pose, which consists of location (t_x, t_y, t_z) and attitude, which is commonly expressed using Euler angles for yaw(ψ), pitch(θ), and roll(ϕ). We can divide these six pose parameters into two sets of three parameters. The first group, which we will refer to as *aviation parameters* (pitch, roll, and altitude), are required to perform basic autonomous maneuvers, while the second, which we term *navigation parameters* (x, y, yaw), are additionally required for autonomous navigation. Current MAV systems [1], [22], [2], [23] carry a simple Inertial Navigation System (INS) consisting of accelerometers, rate gyros, and pressure sensors. These sensors can provide only

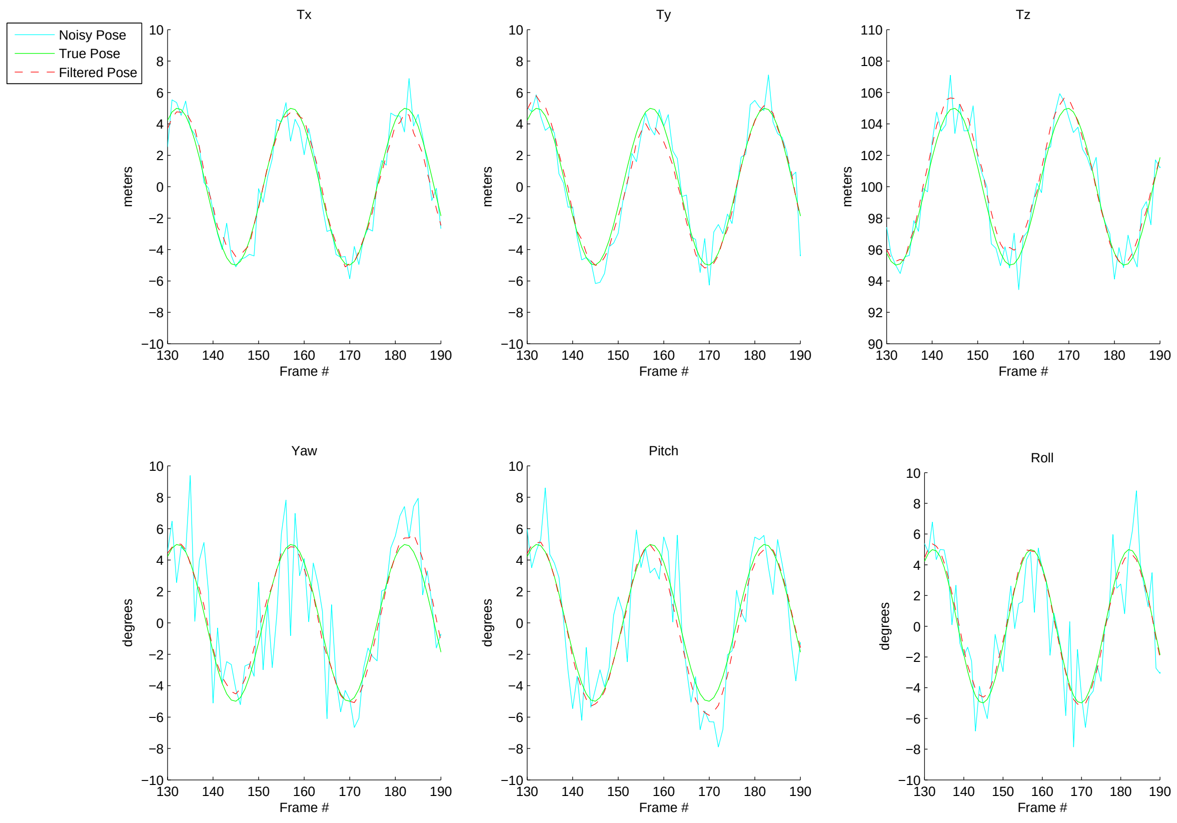


Fig. 2. Simulation results showing actual pose values, noisy pose values and filtered pose estimates (dashed line). The filter was able to significantly improve pose estimates in the presence of noise from both the GPS/IMU and the feature tracker.

relative information about navigation parameters[2]; thus INS based estimates of navigation parameters will drift without bound over time. Because MAVs use low-cost, lightweight MEMS-based inertial sensors, errors in navigation parameter estimates from the INS alone typically increase extremely rapidly, often becoming unacceptably large within a few seconds. For this reason, current MAV systems rely on the Global Positioning System (GPS) to provide estimates of the navigation parameters. While GPS does provide bounded-error estimates of geo-location (x,y) and heading, it makes the operation of the MAV dependent on external infrastructure– the network of orbiting GPS satellites. Signals from these satellites can be blocked, both by environmental obstacles (eg. urban terrain) and by deliberate or unintentional jamming [24]. Much effort has therefore been directed at finding ways to reduce the dependence of MAV platforms on GPS.

Vision-based pose estimation techniques are a promising way to estimate pose in GPS-denied environments, and thus reduce dependence on GPS. Vision sensors are typically already available on MAV platforms, and provide a rich source of information about the environment. There are two main methods of performing vision-based pose estimation reported in the literature. Visual Simultaneous Localization And Mapping, or Visual SLAM (e.g. [25], [26], [27], [28], [29]) is perhaps the most elegant and complete method. SLAM algorithms in general estimate both robot state and the location of landmarks in the environment simultaneously. If perfected, a solution to the Visual SLAM problem would allow a robot to function in a truly autonomous manner, using vision and other sensors to navigate an unfamiliar environment as a human being can, without relying on fiducial markings, GPS signals, or other external infrastructure. However, there are still a number of problems with Visual SLAM which make its practical application challenging. SLAM algorithms in general have non-constant computation time as more and more landmarks are observed, and managing and reducing this computational load is still a focus of ongoing research. Current Visual SLAM systems can use either a video camera alone (e.g. [29]) or a video camera with low-quality MEMS-based inertial sensors (e.g. [28]), and can provide impressive navigational accuracy and stability. Unfortunately, they rely on the assumption that the environment is bounded and relatively small, so that an excessive and growing collection of landmarks does not slow down the processor. This assumption is not valid for MAV platforms, which must navigate in extremely large unexplored environments. Furthermore, in navigation applications, the

[2]As we shall see, INS sensors can provide absolute measurements of aviation parameters. Exploitation of this fact is a critical component of this section.

landmark locations are typically not of interest, meaning that much of this computational burden, while providing greater accuracy, does not contribute directly to the desired result.

In this work, our proposed scenario is a system in which GPS is typically available, but may drop out for an intermediate length of time (several tens of seconds). Our goal, then, is not to produce estimates that do not accumulate error, but estimates that accumulate error slowly enough that they are still accurate within this time window. Visual Odometry (VO) methods are a viable means of performing vision-based pose estimation under this scenario. Named in analogy to wheeled-robot odometry, VO methods use computer vision algorithms to estimate the relative orientation between image frames. With any VO method, then, the key goal is to somehow decrease the amount of error introduced at each step, thereby slowing the growth of error sufficiently that pose estimates are valid within a desired time window.

In this work, we employ two strategies to improve the error characteristics of VO. First, we utilize a novel VO system based upon prior work by Dellaert et al. [30], [31]. Our VO system uses a direct image registration algorithm, estimating a single parametric transformation mapping pixels in an image to ground locations. This is in contrast to more standard feature-based image registration methods, which track a series of feature points and use their motion to infer the relative pose between frames. Direct registration is generally able to produce more accurate results than such feature-based methods; this increased registration accuracy helps to slow the accumulation of pose error.

The second method we use to slow VO error growth is to fuse VO measurements with INS data in an EKF framework. In the literature, a technique known as vision-assisted inertial navigation [32], [33], [34], [35], [36], [37, references therein] is the usual method of doing this. Current techniques use the INS to provide relative measurements of the pose parameters, and these relative measurements are integrated in the EKF time update step to provide absolute pose. The relative pose measurements provided by VO are used in the EKF measurement update step to correct drift in this INS pose estimate. SLAM methods which incorporate inertial sensors similarly use inertial measurements in the time update (e.g. [25], [28]). In this work, we fuse INS and vision in a different way: rather than performing vision-assisted inertial navigation, we propose to perform inertially-aided visual odometry. The distinction is somewhat subtle, but nevertheless significant. One key contribution of this work is that we interpret INS measurements in a different way than the standard vision-aided inertial navigation literature, allowing us to make use of a low-quality MEMS-based INS without inducing INS integration error. This is in contrast to most vision-assisted inertial navigation literature, where the assumption is usually made that a relatively high fidelity INS system is available, or else that another sensor can slow or stop the drift induced by a low quality INS. Another interpretation is possible, however: the three-axis accelerometers in the INS measure the acceleration of the aircraft in each dimension, which for a fixed wing aircraft, will primarily measure the direction of the gravity vector. This information allows us to directly compute the pitch and roll of the aircraft [23]. INS data has been interpreted in this way in the computer vision community to estimate the pose of a camera [38], [39] as well as to perform such tasks as scene reconstruction and camera calibration [40], [41], [42]. Combining these pitch and roll estimates with the altitude estimate obtained from the INS pressure sensors, we have a complete estimate of the aviation pose parameters of the aircraft. Many current MAV systems interpret INS data in this way [1], [22], [2]: however, to our knowledge ours is the first work to apply this information to MAVs in a VO setting. Because we interpret the INS data as providing absolute measurements of the aviation pose parameters of the aircraft, we use INS data in the EKF measurement update, rather than as a time update. Since VO measures the relative pose between frames, we use VO measurements in the time-update step. Thus, VO is the main means of estimating aircraft pose, and its accuracy is improved by incorporating measurements of the aviation parameters from the INS. Use of VO as the time-update step in an EKF framework requires that we be able to propagate both the MAV pose and its covariance matrix. The standard EKF method for doing this (linearizing the time update function) will not work with our proposed VO system, as it is neither differentiable nor available in closed form. The second major contribution of this section is thus a novel means of estimating uncertainty associated with our VO estimates.

In the remainder of this section, we first give a general overview of our pose estimation system (Section III-A), and then describe our VO method and the underlying direct image registration method (Section III-B), giving further background on existing VO methods. We then develop our proposed method of uncertainty propagation through the VO system (Section III-C). Finally, we will present MAV flight results obtained using our method (Section III-D) with a discussion of the computational requirements of this algorithm.

A. Proposed System Overview

The operation of our pose estimation framework is summarized in Figure 4. The goal of this system is to estimate the pose of the aircraft, which we represent with a 6-vector χ:

$$\chi = \begin{bmatrix} t_x & t_y & t_z & \psi & \theta & \phi \end{bmatrix}^T,$$

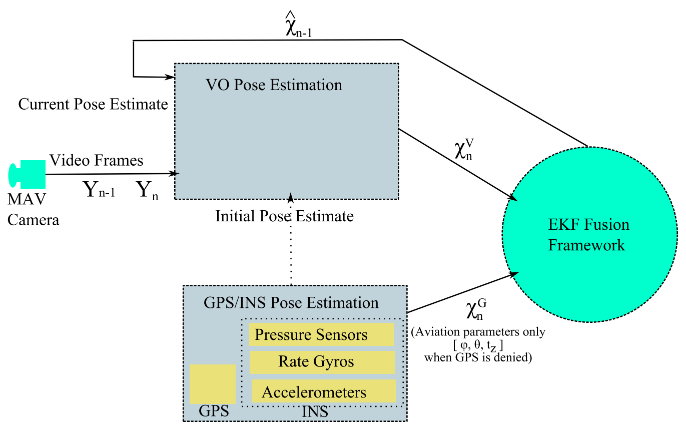


Fig. 4. Layout of our GPS/INS/VO pose estimation system

composed of a 3-D location state (t_x, t_y, t_z) and three Euler angles(ψ, θ, ϕ) representing attitude (yaw[3], pitch, and roll). Video frames Y_n and Y_{n-1} from an MAV camera are fed into the VO system, along with current pose estimates. The VO system then produces an estimated pose χ_n^V for each new video frame Y_n, using the image data Y_{n-1} and estimated pose $\hat{\chi}_{n-1}$ of the previous frame.

Independently, information from other on-board sensors are fed into the GPS/INS pose estimation system, which separately estimates a pose χ_n^G of the aircraft at the time each frame was taken. If GPS is unavailable, this estimate includes only the aviation parameters, which can be obtained from the INS alone. In our system, altitude is computed directly from the autopilot pressure sensors, while pitch and roll are computed by combining accelerometer and rate gyro measurements in a complimentary filter. Pitch/roll states are propagated forward in time using rate measurements from the gyroscopes. Accelerometer measurements are used to give bounded error estimates of pitch and roll, computed according to the following formula:

$$\begin{aligned}
\left[\begin{array}{ccc} A_x & A_y & A_z \end{array}\right] &\equiv \text{accelerometer readings} \\
\phi_{acc} &= \tan^{-1}\left(\frac{A_y}{-A_z}\right) \\
\theta_{acc} &= \tan^{-1}\left(\frac{A_x}{-A_y \sin(\phi) - A_z \cos(\phi)}\right).
\end{aligned}$$

The weighted sum of the accelerometer based measurements and predicted states become the new pitch/roll estimates. If the aircraft is turning, the accelerometers will measure not only the lift force opposing gravity, but also extra acceleration from the d'Alembert force due to centripetal acceleration. To help account for this fact, the relative weight of the accelerometer-based measurements is reduced as the turn rate of the MAV increases, causing the system to rely more heavily on the propagated values. In our experience, this method produces sufficiently accurate estimates to enable MAV navigation (see [23]).

Because these INS based pose measurements have bounded error as we have discussed, we model this estimate as the true pose of the aircraft corrupted with zero-mean Gaussian noise (ν):

$$\text{GPS unavailable: } \chi_n^G = \left[\begin{array}{ccc} t_z & \theta & \phi \end{array}\right]^T + \nu. \quad (26)$$

We desire to fuse these partial measurements of aircraft pose with the pose information from the VO system using an EKF framework. The standard EKF framework estimates the state of a system given knowledge of the system dynamics and measurements that are functions of the state:

$$\chi_n = \mathbf{f}(\chi_{n-1}, u_n) + \eta \quad (27)$$
$$y_n = \mathbf{h}(\chi_n) + \nu \quad (28)$$

where η and ν are zero-mean, Gaussian random vectors with covariance matrices Q and R respectively. At each time step, we estimate the new state of the system and its covariance from the previous state:

$$\hat{\chi}_n^- = \mathbf{f}(\hat{\chi}_{n-1}, u_n) \quad (29)$$
$$\hat{P}_n^- = F\hat{P}_{n-1}F^T + Q \quad (30)$$

where the matrix F is the jacobian of the system dynamics function:

$$F = \left.\frac{\partial \mathbf{f}}{\partial \chi}\right|_{\chi=\hat{\chi}_{n-1}, u=u_n}. \quad (31)$$

When a measurement becomes available, we incorporate the information it provides about the state, performing what is known as a measurement update step:

$$\hat{\chi}_n = \hat{\chi}_n^- + K\left(y_n - \mathbf{h}\left(\hat{\chi}_n^-\right)\right) \quad (32)$$
$$\hat{P}_n = (I - KH)\hat{P}_n^- \quad (33)$$

where H is the jacobian of the measurement function (analogous to F):

$$H = \left.\frac{\partial \mathbf{h}}{\partial \chi}\right|_{\chi=\hat{\chi}^-} \quad (34)$$

As we have discussed, we use the aviation parameters available from the INS as our measurement function. This means that our $\mathbf{h}$ function is in fact just a linear operator, and we can find H directly:

$$\begin{aligned}
y = \begin{bmatrix} \theta \\ \phi \\ t_z \end{bmatrix} &= \mathbf{h}(\chi) \\
&= \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}}_{H} \chi. \quad (35)
\end{aligned}$$

[3]Yaw is typically defined as the compass direction in which the nose of the aircraft is pointing, while heading is the direction in which the aircraft is moving. If the aircraft is flying with a 'crab angle' (i.e. the nose is not pointing exactly in the direction of flight) due to wind conditions, these two quantities will not be identical. Pose estimates obtained using VO provide yaw information, while heading estimates can be provided directly by GPS. Heading can also be estimated by using the difference in location estimates.

By contrast, the time update in our EKF will come from VO, meaning that our $\mathbf{f}$ function represents the VO system as follows:

$$\begin{aligned} \chi_n^- &= \mathbf{f}\left(\chi_{n-1}\right) \\ &= \text{vo\_system}\left(\chi_{n-1}, Y_{n-1}, Y_n\right). \end{aligned} \tag{36}$$

In the next section, we will describe the operation of our VO pose estimation system, which is represented by the function $\mathbf{f}(\cdot)$. We will then proceed in section III-C to approximate F, enabling us to implement equation (30).

B. Visual Odometry System

Several VO frameworks are delineated in the literature. Most commonly used existing methods function by detecting and tracking feature points between frames in a video sequence, and using the motion of these points to estimate the relative pose between frames. This is done by using feature point motion to estimate either the essential matrix [43], [44], [45], [46], [47] or a homography [48], [49] relating pairs or sets of frames, and then decomposing these matrices [50], [51] to find the relative pose.

All VO methods share a common implementation challenge that must be addressed to allow absolute pose to be estimated. This problem is that of determining the scale factor of the estimated relative pose. Because a video camera is a bearing-only sensor and provides no depth information, it is impossible to distinguish whether a pair of frames are widely separated and observing large, distant objects or closely spaced and observing small, nearby objects. If care is not taken to ensure that relative pose estimates are expressed in the same scale then gross errors in absolute pose estimates can be accumulated very rapidly. This problem is typically addressed in the literature by triangulating the 3-D location of feature points common between two frame pairs.

In this work, we propose a novel VO strategy, based upon prior work by Dellaert et al. [30], [31]. Rather than infer the inter-frame relative pose by using the motion of extracted feature locations, we directly compute the absolute pose of the second frame from the absolute pose of the first frame by means of an iterative image registration approach. This approach works by projecting the first observed image onto the terrain and rendering a view of this projected data from the currently estimated pose of the second frame. The estimated pose of the second frame is iteratively adjusted by means of image jacobians to make the second frame and the re-projected first frame match as closely as possible. Thus, this method directly estimates a single parametric transform using the captured image as a whole, rather than the estimated motion of selected feature points. This fact typically allows direct image registration methods to provide greater accuracy in image registration. Furthermore, since the absolute pose of each frame is estimated, the scale factor problem is handled implicitly.

Direct image registration such as we are performing depends upon three main assumptions: (1) that the scene

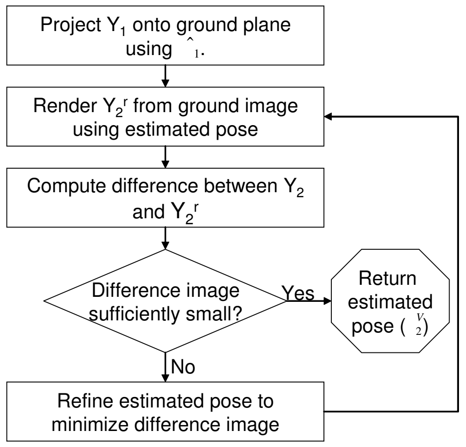

Project Y1 onto ground plane using ^ 1.
Render Y2r from ground image using estimated pose
Compute difference between Y2 and Y2r
Difference image sufficiently small?
Yes
No
Return estimated pose (2)
Refine estimated pose to minimize difference image

Fig. 5. Our method for computing the pose of the MAV when Image 2 was captured assuming Image 1's pose is perfectly known.

being imaged is planar, (2) that all image motion is due to camera motion, meaning that motion due to independently moving objects is negligible, (3) there is sufficient texture in the imaged scene to allow the iterative descent registration algorithm to avoid converging to a local minimum.

While the planarity assumption is not appropriate for ground-based robots, it is generally a reasonable assumption in other applications. Unmanned Underwater Vehicle applications [52], [53], [54] commonly make this assumption. As MAVs are typically relatively distant from the terrain they observe, this assumption is often workable in many fixed-wing MAV situations. In addition, we have found that our direct registration algorithm is robust to small amounts of non-planarities (e.g. trees, small structures) in the images. This is a reasonable scenario for many applications; low flights in complicated (e.g. urban) terrain will violate this assumption, however.

The assumption that objects do not move independently is not problematic in most environments. At typical MAV altitudes, any moving objects will occupy only a tiny fraction of a captured image, and thus will have little effect on image registration. This assumption will fail in some cases (i.e. viewing a highway with heavy, fast-moving traffic) but will be a good assumption in many others.

Sufficient image texture is also usually a good assumption: most UAV flights are daytime flights, and most real flight environments contain significant visual texture. Certain flight environments could of course cause the direct image registration to not converge. In general, however, our practical experience suggests that direct registration is more robust than feature-based methods in low-texture video.

Our VO algorithm for estimating the current pose of the MAV is illustrated in Figure 5. To estimate the pose of video frame Y_2 (or rather, the pose of the aircraft when this frame was captured), we assume that a previous frame Y_1 is available with associated pose information $\hat{\chi}_1$ for that frame. We also assume that we have a coarse estimate of the pose from which Y_2 was captured $\left(\chi_2^E\right)$. This coarse estimate could be obtained from the current GPS/INS estimate χ_2^G, the pose of the previous frame $\hat{\chi}_1$, or the result of a quick feature-based motion estimation algorithm. When registering sequential video frames (30 fps frame rate), simply using the pose of the previous frame as the initial pose estimate $\left(\text{i.e. let } \chi_2^E = \hat{\chi}_1\right)$ was found to produce the most rapid convergence, as the aircraft typically did not move far enough in one frame interval to make this a bad initial guess. The goal of our algorithm is to compute a more refined estimate χ_2^V of the MAV pose when frame Y_2 was captured.

The first step shown in Figure 5 is to project Y_1 onto a ground image. The projection process assumes that the terrain over which the MAV is flying is planar and horizontal and uses the estimated pose $\hat{\chi}_1$ with respect to this ground plane to produce an ortho-rectified image of the region of ground observed by Y_1. This projection is computed by perspective warping: i.e. the ground image is a perspectively warped version of the captured image. Rather than interpolating between pixel values, a gaussian point spread function is assumed to act on each pixel value, and the perspective projection of this point spread function determines the amount by which each pixel in Y_1 affects each ground image pixel. Further explanation and details of this warping process are given by Dellaert et al. [30], [31].

Once we have inferred the appearance of the ground plane using image Y_1, we desire to iteratively refine our initial pose estimate for Y_2. At the kth iteration, we produce a rendered image $Y_2^r(k)$ of this ground image using the current pose estimate $\chi_2^V(k)$ for image Y_2. This rendering process is simply the inverse of the projection process, and is performed using a projectively distorted gaussian point spread function to determine the amount by which each ground pixel affects a given pixel in $Y_2^r(k)$. The rendered image $Y_2^r(k)$ represents the visual information in Y_1 as it would appear in Y_2, assuming that the poses $\hat{\chi}_1$ and $\chi_2^V(k)$ used for projection and rendering were accurate. The difference or residual image $\left(Y_2^r(k) - Y_2\right)$ provides information about the error in the current pose estimate $\chi_2^V(k)$. To determine an update $\Delta\chi_2^V(k)$ to the current pose estimate, we use a variant of the popular Lucas-Kanade image registration method [55], based on Gauss-Newton gradient descent. We attempt to choose $\Delta\chi_2^V(k)$ to minimize the pixel for pixel squared magnitude of the residual image:

$$J = \sum_{p \in \mathbb{P}} \left(Y_{2,p} - Y_{2,p}^r\right)^2 \tag{37}$$

where $\mathbb{P}$ is the set of all pixels in image Y_2, $Y_{2,p}^r$ are the pixels in the rendered image and $Y_{2,p}$ are the pixels in Image 2. A Gauss-Newton iteration essentially consists of computing partial derivatives of the residual image with respect to all of the pose parameters. Each of these partial derivative images approximates the change in the residual image caused by a differential change in the associated pose parameter. The goal at each iteration is to express the residual image as a weighted sum of the different Jacobian images, after which the pose is changed according to these weights. A graphical example of a single iteration is shown in Figure 6.

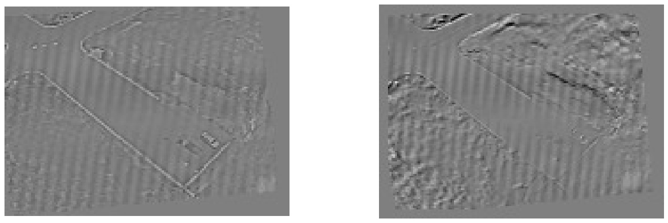


Fig. 7. Residual image differences produced by our direct registration method (left image) and a standard feature-based registration method (right image). Feature-based registration was performed on a 640 × 480 video sequence, tracking Harris corners between frames using the OpenCV™toolbox, using RANSAC to estimate a homography matrix relating each frame pair, and warping the first frame to align it with the second. Direct registration was performed using the method indicated in this section on a 4× downsampled version of the same video. Direct registration is able to consistently reduce the minimum mean squared pixel error compared to feature based registration.

As discussed in [30], [31], the partial derivatives or "Jacobian Images" of the residual can be approximated using the chain rule as follows:

$$\underbrace{\frac{\partial Y_2^r}{\partial \square}}_{\text{Jacobian Image}} = \underbrace{\frac{\partial Y_2^r}{\partial x}}_{\nabla_x} \frac{\partial x}{\partial \square} + \underbrace{\frac{\partial Y_2^r}{\partial y}}_{\nabla_y} \frac{\partial y}{\partial \square} \tag{38}$$

Each of the terms in this equation represents an "image" or matrix of values, one for each pixel location. The symbol $\square$ represents one of the six pose parameters $[t_x, t_y, t_z, \psi, \theta, \phi]$, and the terms labeled ∇_x and ∇_y are gradients of the rendered image, i.e. partial derivatives of the luminance function in the vertical and horizontal image directions. The $\frac{\partial y}{\partial \square}$ and $\frac{\partial x}{\partial \square}$ terms represent the differential location change of the image of the preimage of each image point. That is, each pixel location (x, y) is imaging a particular world point P, and a differential change in any pose parameter $(\square)$ will cause a differential change in the (x, y) image coordinates of the projection of P. The $\frac{\partial y}{\partial \square}$ and $\frac{\partial x}{\partial \square}$ terms thus represent the way in which a feature observed at any point in the image will appear to move due to a differential change in the pose parameter $\square$. After multiple iterations like the one shown in Figure 6, the estimated change $(\Delta\chi_2^V(k))$ in the pose estimate will become very small. At this point, the current pose estimate $\chi_2^V(k)$ becomes the final χ_2^V returned from our VO algorithm.

Figure 7 demonstrates the potential improvement in registration given by direct registration methods versus more standard feature-based image registration. Since more accurate pixel registration implies more accurate relative pose estimates, improved accuracy can lead to slower growth in VO pose estimates, leading to more accurate overall pose estimates.

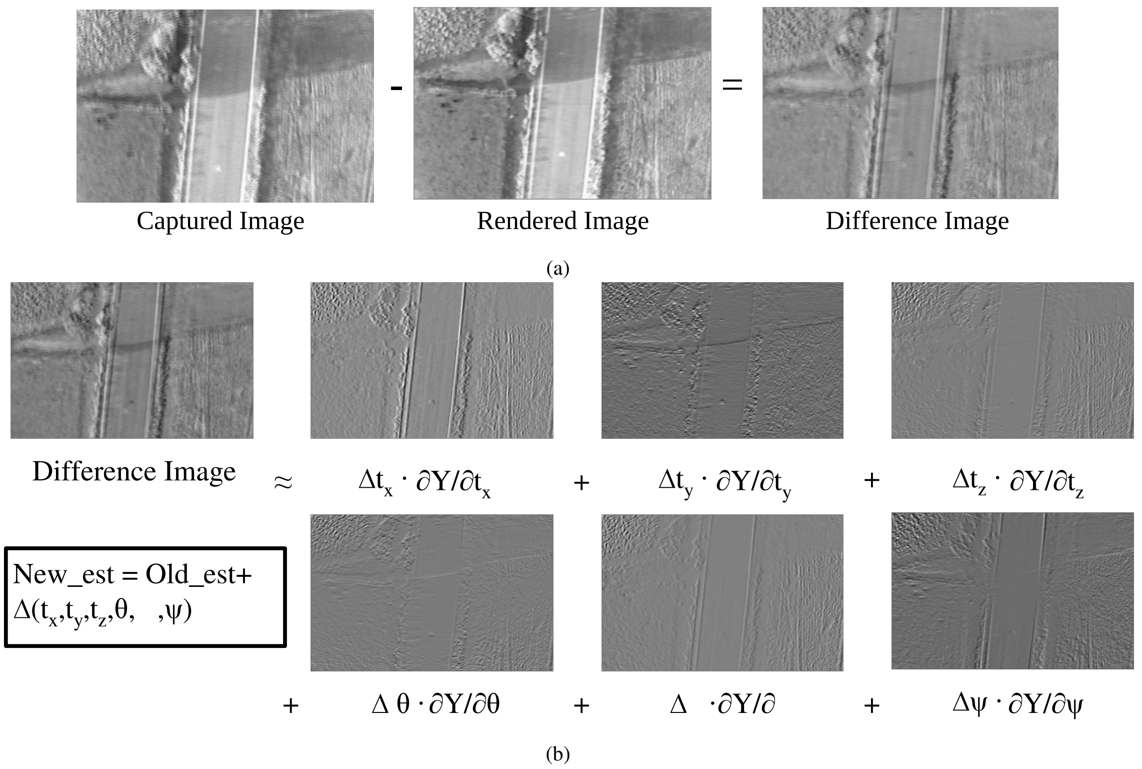


Fig. 6. An example iteration of the image registration process using our Gauss-Newton registration method. In subfigure (a), a difference image is created to evaluate how accurate the current pose estimate is. In subfigure (b), the new pose (New_est) is computed using the difference image and the Jacobian images.

C. VO Covariance Estimation

As discussed in Section III-A, to fuse INS and VO measurements in an EKF framework, we need to be able to find the matrix

$$F = \left.\frac{\partial \mathbf{f}}{\partial \chi}\right|_{\chi=\hat{\chi}_{n-1}}$$

that linearly approximates the state transition function. The state transition function $\mathbf{f}$ in our EKF is the VO system described in the previous section, which produces the current aircraft pose χ_2^V given the estimated previous pose $\hat{\chi}_1$. Unfortunately, this $\mathbf{f}$ function is not differentiable, and is expressed only as an iterative algorithm, not in mathematical closed form. In this section, we will make some simplifying assumptions that allow us to approximate the jacobian F of this algorithm. This will allow us to propagate covariance in pose $\chi_1$to covariance on pose χ_2^V, enabling the fusion of VO and INS measurements.

The final χ_2^V produced by our VO system is a function of both the pose $\hat{\chi}_1$ and the two images Y_1 and Y_2. The interplay of these two images in the iterative image registration algorithm leads to a non-differentiable $\mathbf{f}$. Stated differently, changes in χ_2^V can be due either to changes in the quality of image registration between Y_1 and Y_2 or due to changes in the original pose $\hat{\chi}_1$: this fact makes it impossible to differentiate $\mathbf{f}$ directly. Even if we could parametrize and precisely describe "the quality of image registration" in a meaningful way, differentiating $\mathbf{f}$ with respect to this parametrization would still involve differentiation of an iterative procedure. To rectify this situation, we will first assume in Section III-C1 that the image registration process is able to perfectly register Y_1 with Y_2: this is the same as assuming that all our uncertainty about the final pose χ_2^V is due to propagated uncertainty in $\hat{\chi}_1$. We will find the desired matrix F using this assumption. We will then discuss error due to mis-registration in Section III-C2.

1) Propagating Errors in Image 1 Pose to Image 2: In order to find the component of uncertainty on χ_2^V (due to propagation of uncertainty from $\hat{\chi}_1$), we need to be able to characterize the function $\mathbf{f}(\cdot)$ such that $\chi_2^V = \mathbf{f}(\hat{\chi}_1)$. We seek to linearize this function so that we can perform a standard linear covariance update $P_2^V = F\hat{P}_1F^T$. To begin, we note that once the iterative image registration process has converged, $\hat{\chi}_1$ and χ_2^V can each be used to compute the homography matrices H_{G1} and H_{G2}, which map pixel locations in images Y_1 and Y_2 respectively to locations on the ground image. These homographies were used respectively in the projection and rendering processes of the VO system (see Section III-B), and are thus available from the VO process. H_{G1} and H_{G2} individually have uncertainty associated with them, because each of their components is a function of $\hat{\chi}_1$ and χ_2^V, which are imperfectly known and thus have associated covariances. Because homography matrices can be composed by matrix multiplication and are invertible [56],

[51], we can combine these two homography matrices into a single homography that maps pixel locations in Y_1 to pixel locations in Y_2 as:

$$H_{12} = H_{G2} H_{G1}^{-1} = H_{G2} H_{1G} \tag{39}$$

If, as we have assumed, the registration between Y_1 and Y_2 is perfectly accurate, then this homography H_{12} is perfectly known. Thus, although both H_{G2} and H_{1G} (the inverse of H_{G1}) have associated uncertainty, their product H_{12} does not. This observation is a direct consequence of the fact that VO methods fundamentally measure relative poses: while χ_1 and χ_2 may both be incorrect, the accuracy of the relationship between them is constrained only by the accuracy of image registration.

The relationship in equation 39 forms the basis of our desired $\mathbf{f}(\cdot)$. We first post-multiply by H_{G1}:

$$H_{G2} = H_{12} H_{G1} \tag{40}$$

Our insight about H_{G1},H_{G2}, and H_{12} allows us to write:

$$H_{G2}(\chi_2) \quad = \quad H_{12} H_{G1}(\chi_1)$$

In order to isolate χ_2 as a function of χ_1 from this last equation, it would be desirable if we could invert the function $H_{G2}(\chi_2)$; we would like to be able to deduce the pose of a camera given a homography mapping its image points to the ground. We will refer to this inverse function as ξ instead of H_{G2}^{-1}, to emphasize the fact that the inverse we would like is not the matrix inverse of the matrix H_{G2}, but rather a function mapping a homography matrix to a pose. If we could find such a function, we would have our desired formula for $\mathbf{f}$:

$$\begin{aligned} \xi\left(H_{G2}\left(\chi_2^V\right)\right) &= \xi\left(H_{12} H_{G1}\left(\hat{\chi}_1\right)\right) && (41) \\ \chi_2^V &= \xi\left(H_{12} H_{G1}\left(\hat{\chi}_1\right)\right) && (42) \\ \chi_2^V &= \mathbf{f}\left(\hat{\chi}_1\right) && (43) \end{aligned}$$

We could then compute the desired derivative F by the chain rule:

$$\frac{\partial \mathbf{f}}{\partial \hat{\chi}_1} = \frac{\partial \chi_2^V}{\partial \hat{\chi}_1} \quad = \quad \frac{\partial \xi}{\partial H_{G2}} \frac{\partial H_{G2}}{\partial H_{G1}} \frac{\partial H_{G1}}{\partial \hat{\chi}_1} \tag{44}$$

The function $H_{G1}(\hat{\chi}_1)$ can already be computed in closed form using standard computer vision techniques, and its derivative, the term $\frac{\partial H_{G1}}{\partial \hat{\chi}_1}$ in equation 44 is a 9×6 matrix of partial derivatives that can be computed from this closed form expression in a straightforward manner. The term $\frac{\partial H_{G2}}{\partial H_{G1}}$ is a 9×9 matrix of partial derivatives: these partials are also straightforward to compute, as they are simply elements of H_{12} (see equation 40).

The only remaining obstacle is the first term in equation 44, determining the derivative of the ξ function. As discussed in [51], there are well documented methods of decomposing a homography matrix to determine a relative pose; these methods, however, in general produce four solutions among which we must choose based on the cheirality constraint. This fact makes the ξ function non-differentiable. Instead of attempting to differentiate ξ directly, we notice that we can approximate changes in the homography matrix H_{G2} due to changes in χ_2^V using the partial derivative matrix $\frac{\partial H_{G2}}{\partial \chi_2}$, just as we do for H_{G1}. The inverse of this linear mapping, if it existed, would give changes in χ_2^V due to changes in H_{G2} as desired. The inverse does not exist, as the 9×6 matrix $\frac{\partial H_{G2}}{\partial \chi_2^V}$ represents an over-determined system. We can, however, find the pseudo-inverse of the matrix $\frac{\partial H_{G2}}{\partial \chi_2}$, which still maps changes in H_{G2} to changes in χ_2, minimizing error in the elements of H_{G2} . We use this pseudo-inverse to approximate the derivative $\frac{\partial \xi}{\partial H_{G2}}$.

Combining these three terms, we compute the 6×6 matrix of partial derivatives $\frac{\partial \chi_2^V}{\partial \hat{\chi}_1}$ as:

$$\begin{aligned} \frac{\partial \chi_2^V}{\partial \hat{\chi}_1} &= \left(Q^T Q\right)^{-1} Q^T \frac{\partial H_{G2}}{\partial H_{G1}} \frac{\partial H_{G1}}{\partial \hat{\chi}_1} && (45) \\ & \text{where} \\ Q &= \frac{\partial H_{G2}}{\partial \chi_2^V} && (46) \end{aligned}$$

We now have the desired partial derivative matrix of our VO method, which we can use to approximate covariance propagation.

2) Determining covariance with imperfect registration: Naturally, the process of determining H_{12} (i.e. registration) is not, as we assumed in the previous subsection, without error. This registration error will cause the difference image to have residual structure (which is not due to parallax) after the registration process completes. We address this source of error by computing the ratio of the residual cost function J of equation 37 with an empirically determined cost value, and boosting the diagonal values of the covariance matrix P_2^V proportionally. Doing this is similar to the "Q-boosting" technique common in EKF practice: we simply increase our estimated uncertainty on the VO pose such that the overall filter yields desirable results. In practice, this extra boost in the diagonal elements of P_2^V was not found to be necessary to yield good results, and was not used in generating result data.

D. Results and Analysis

In order to evaluate our pose estimation framework, we estimate MAV pose without using GPS information, and show that fusion of VO and INS information allows MAV location to be estimated with accuracy similar to that of GPS for a reasonable period of time.

All results presented in this work are collected using an inexpensive, hand-launchable MAV platform, shown in Figure 8. The aircraft is a flying wing design, with a 6-foot wingspan, constructed of EPP foam. The on-board Kestrel$^{\text{TM}}$autopilot and associated Virtual Cockpit$^{\text{TM}}$software platform allow the aircraft to autonomously aviate and navigate. The Kestrel$^{\text{TM}}$autopilot comprises a small microcontroller and a collection of sensors that includes three-axis accelerometers, rate gyroscopes, and differential pressure sensors. Pitch, roll, and altitude are estimated on-board from these sensor readings, while 2-D location and heading are estimated using a small on-board GPS receiver [1], [23]. This pose estimate and other telemetry data is transmitted to a ground station at a rate of about 4 Hz. A separate camera/transmitter system collects video footage during flight and transmits this video to the ground station. This video stream is synchronized on the ground station with the stream of pose estimates from autopilot telemetry. VO and sensor fusion are performed off-line using this data.

1) Pose Estimation during GPS dropout: A key goal of this work is to explore the use of VO for MAV localization during GPS dropout. In order to evaluate the accuracy of our fused VO/INS pose estimates, we compare the estimated (t_x, t_y, t_z) location of the aircraft with baseline location measurements produced by the current MAV autopilot pose estimation method, which uses a GPS receiver to measure t_x and t_y, and a differential pressure sensor to measure t_z. We will hereafter refer to these baseline pose estimates as `gpsins` pose estimates. These baseline location estimates are here compared with location estimates from:

1) Our unaided VO system (referred to as `voonly`) shown in Figure 9
2) Our fusion system, incorporating only VO and INS measurements (referred to as `voins`) shown in Figure 10.

In each of these figures, we display the path of the MAV as estimated by `gpsins` and by one of the vision-aided fusion schemes, giving both a horizontal and vertical view (subfigures (a) and (b), respectively), and plot the time-varying distance between these two paths (subfigure (c)). Clearly the errors in `voonly` location are both much larger than those of `voins`, and increase dramatically over the course of the flight ($\sim$70 seconds). This demonstrates that fusion of VO and INS data can significantly reduce the drift in location estimates that is inherently part of VO systems, as well as dramatically reducing (˜40%) the worst-case location error. It is also meaningful to realize that the cyclic pattern of errors present in Figures 9(c) and 10(c) is likely a consequence of inaccuracies in camera mounting, wind estimation, and temporal data association. It is perhaps remarkable that position can be estimated to within $<$40m of GPS estimates in the presence of these inaccuracies. As the size, cost, and field use constraints on MAV platforms necessarily make them prone to these errors, the ability to function in their presence is an important benefit for MAV platforms. It should be noted that this error is comparable to the error obtained by some SLAM-based visual/inertial navigation solutions, such as that of Kim and Sukkarieh[25] and Bryson and Sukkarieh[26] (whose flight path is similar to ours). Several other SLAM-based solutions give much better accuracy in simulation (e.g. [57]) or in limited environments with relatively rapid loop closure (e.g. [29]).

2) Computational Complexity: The results given here are presented as a proof of concept. The navigational results shown were obtained using real flight data, but processing was performed offline, in a framework that was not optimized for speed. A number of further optimizations are possible, which we feel would allow our estimation framework to run at a frame rate of about 5Hz or better on computing hardware compatible with MAV weight & power constraints (˜1.5 GHz standard laptop-style CPU). These optimizations could include the following:

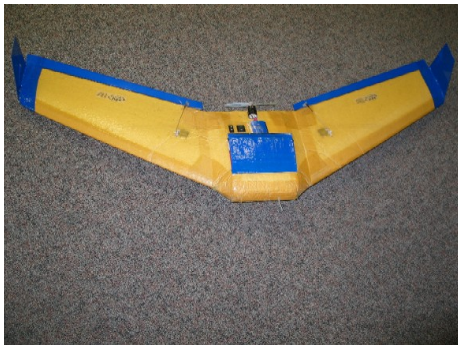
- The simulation environment is currently coded in MATLAB, and no MEX functions are utilized. A functioning system would need a C/C++ implementation, which in itself would bring performance significantly closer to realtime.
- We currently perform forward additive image registration: that is, the second image is repeatedly warped (requiring expensive recomputation of image jacobians) until it matches the first image. Baker et al. [55] present several ways of improving on this, notably the inverse compositional method, which needs to compute jacobians only once and applies the inverse of the computed warping to the first image. This could also significantly reduce computational load.
- Not all pixels need actually be compared during the direct registration process: only a sampling of pixels would be needed. This could significantly reduce computation costs.

Memory requirements associated with this method are both fixed and reasonable, as only the previous video frame and the vehicle state estimates need to be stored.

IV. A reduced-drift approach for visual/IMU-fusion based navigation

As shown in the prior section, there are significant advantages to fusing together data from inertial and visual sensors. However, there are significant computational costs associated with the method from Section III. Similarly, for the most accurate navigation estimate possible, a batch method that analyzes all vision and IMU information from an entire flight was introduced in [58]. While accurate, this method cannot be utilized in real-time due to nature of batch optimization routines. Therefore, the paper also introduces a recursive method which is essentially a SLAM filter. Other implementations of SLAM-based filters for navigation can also be found in [59], [60], [61]. While SLAM-based methods are highly effective, there are two bottlenecks to SLAM that make them difficult to implement in the computationally-limited environments that characterize MAVs. First, visual SLAM requires that objects in the video be tracked for an extended period of time. Second, the size of the state grows with the number of landmarks that SLAM is attempting to find the location for, dramatically increasing the computation time required. In [62], the size of the state is limited, but tracking of features over a long period of time is still required.

While it is computationally expensive to track points for an extended period of time in video, it is relatively simple to track points over a small number of video frames. Therefore,



(a) MAV

(b) Kestrel^{TM}Autopilot

Fig. 8. The MAV platform used in this work. The MAV is a flying wing aircraft constructed of EPP foam with a 6 foot wingspan, controlled by a Kestrel^{TM}autopilot.

we focus in this section on a method that utilizes only the relationship between objects in two frames of video. Other methods that utilize only two frames of video ([63], [64], [65]) have been introduced previously. [64], [65], however, requires that the terrain being observed is planar, while we assume in this section that the points being tracked from frame to frame are not planar (a better assumption for indoor or dense environments that would obscure GPS signals).

In this section, we focus on utilizing the *epipolar constraint* for fusing visual measurements with the IMU as described in [63]. Using the epipolar constraint, however, has three significant weaknesses that must be addressed when performing fusion with IMU measurements for MAV navigation. First, the epipolar constraint biases movements of the camera toward the center of points in the image. Second, visual measurements always include a "scale ambiguity" – it is impossible to distinguish between the camera moving quickly and observing an object that is far away and the camera moving slowly and observing an object that is close. Third, when using vision to navigate, the navigation state of the camera can only be determined relative to its previous navigation states. [4]

In this section, we present methods to overcome each of these three weaknesses in utilizing the epipolar constraint. First, to overcome the bias in the epipolar constraint, we analyze the epipolar constraint equation and propose an alternate algorithm for computing deviations from the epipolar constraint. Second, to overcome the scale ambiguity of vision, we integrate a differential air pressure sensor into the fusion system. On a fixed-wing MAV, the differential air pressure sensor is capable of measuring the airspeed of the MAV. This airspeed can be treated as a direct measurement of the velocity magnitude, allowing the scale ambiguity of vision to be overcome. Third, because vision measurements of motion are relative, we propose using the *minimal* sampling rate at which vision can be effectively fused with IMU data. We demonstrate that sampling at the minimal, rather than maximal, rate increases the accuracy of the overall navigation system. We also discuss limitations on choosing the minimal sampling rate.

Once the weaknesses of epipolar-based fusion are overcome, it is possible to enable on-line estimation of inertial sensor biases. We prove this capability by performing an observability analysis of a simplified system, and demonstrate improved navigation results when estimating biases.

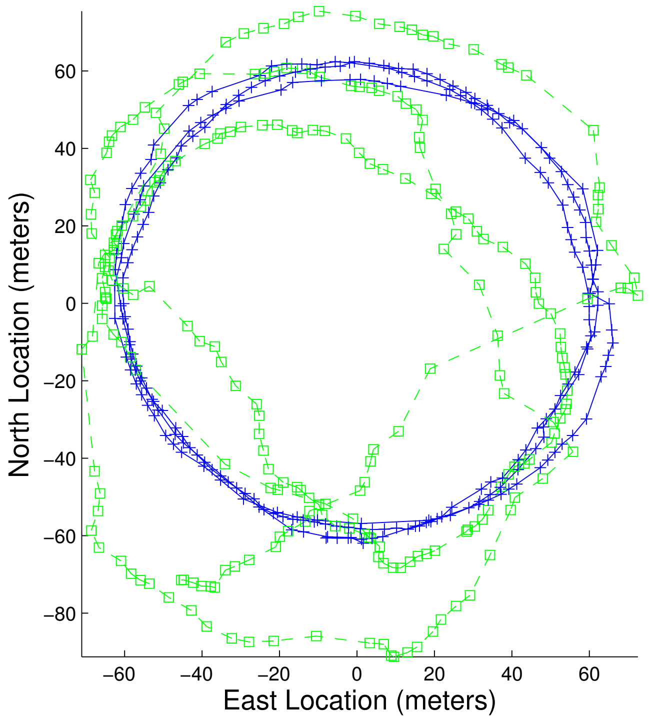
The remainder of this section is organized as follows. In Section IV-A, we discuss the general framework used for fusing vision and IMU information. In Section IV-B, we describe our three modifications for improving the fusion of visual and IMU information. In Section IV-C, we describe our method for estimating the biases of the inertial sensors. Section VII-B demonstrates the improvement in navigation state estimates achieved by implementing our modifications.

A. Epipolar constraint-based fusion

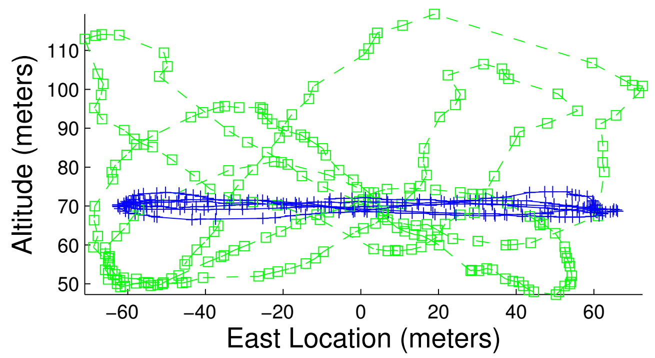
In this section, we first describe the epipolar constraint which is used in our fusion setup. We then describe how the epipolar constraint can be used in a Kalman filter to enable fusion between visual and inertial information.

1) The epipolar constraint: The epipolar constraint can be utilized whenever a single fixed object is observed by a camera at two locations (or two cameras at different locations). Given that any three points in the world form a plane, a single world point and the two camera projection centers form a plane in the 3-D world – the *epipolar* plane. Similarly, when a world point is observed in two images, the

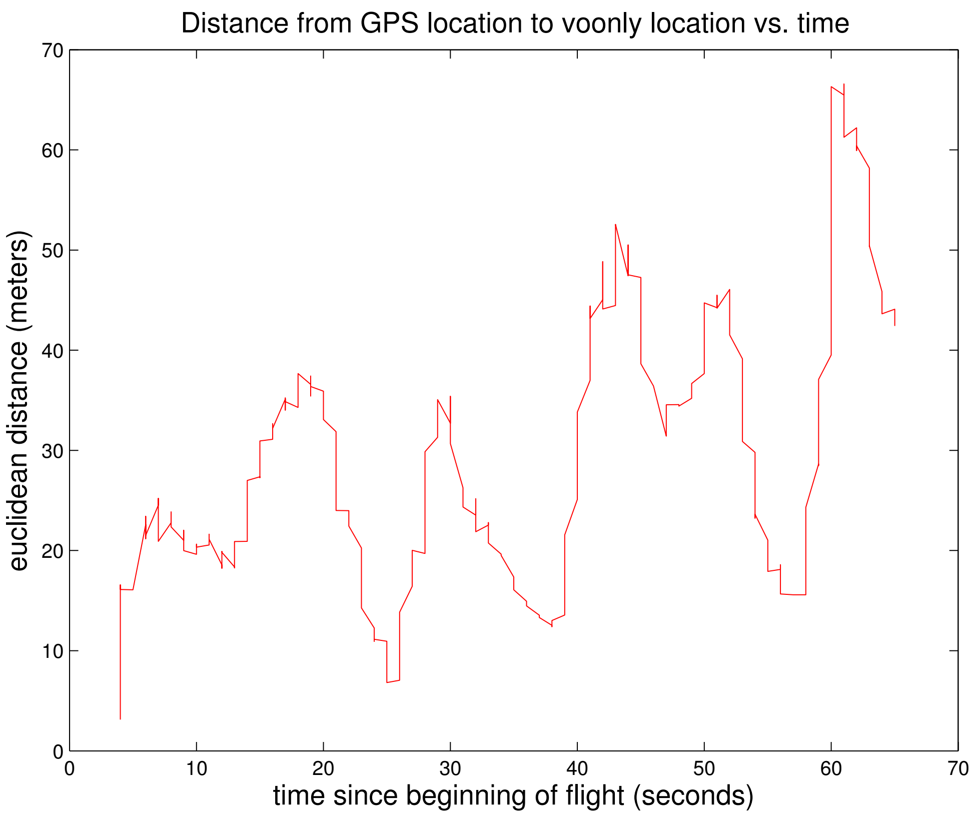
[4]Note that while it is possible to determine absolute position or attitude using vision, knowledge of pre-existing visual "landmarks" is required. We do not address these techniques in this section as we are interested in using MAVs to explore new areas, not fly over pre-mapped areas.



(a)

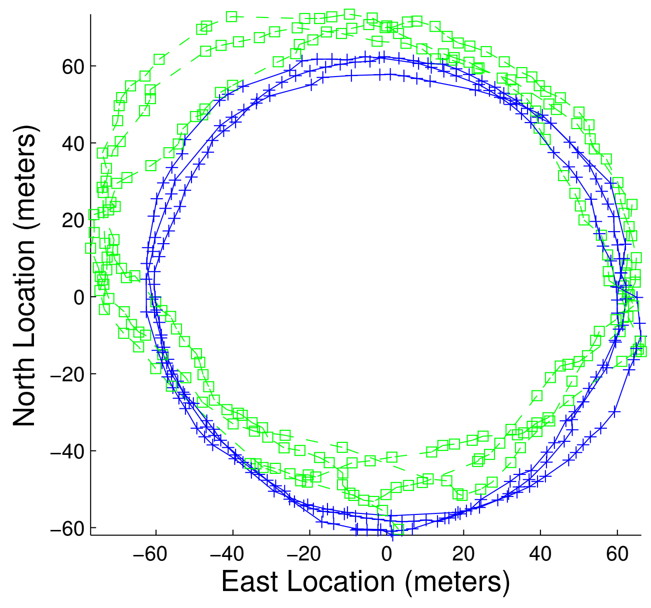


(b)

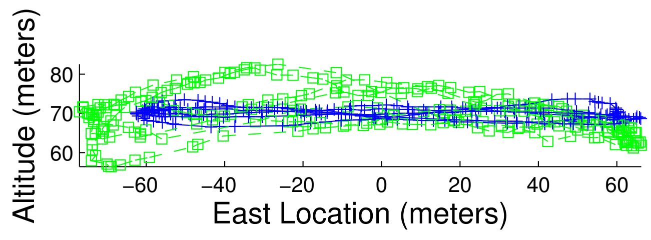


(c)

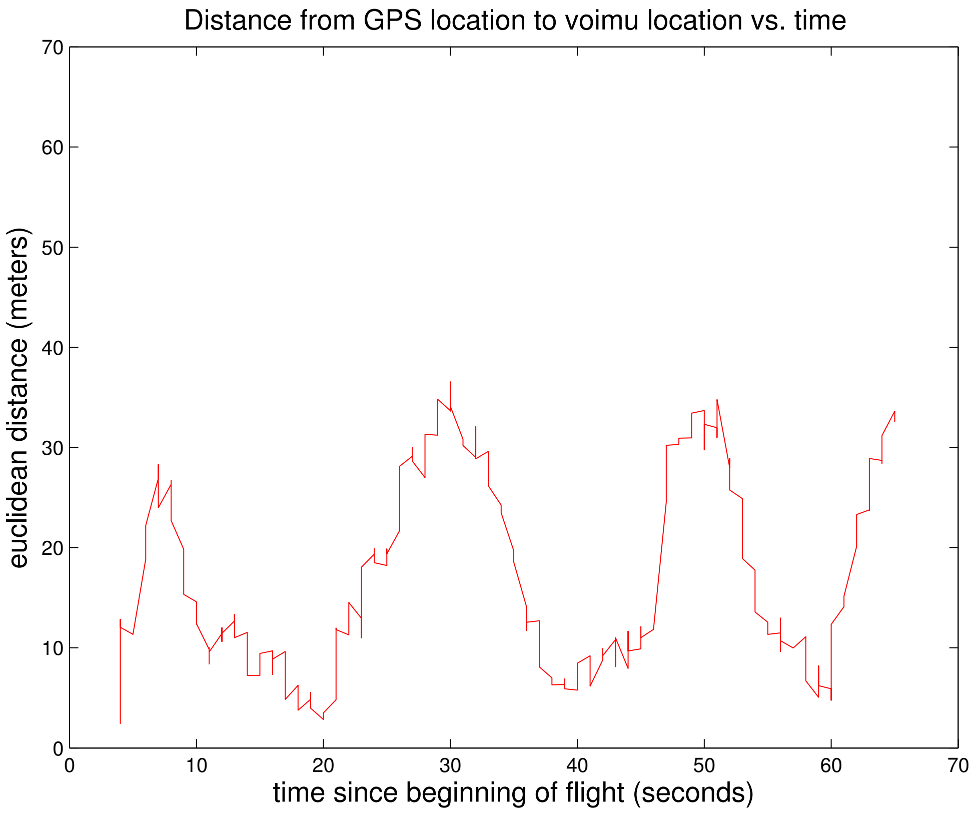
Fig. 9. (a,b) `gpsins` location estimates (blue +) and `voonly` location estimates (green □) in a circular flight path. (c) Euclidean distance between `gpsins` and `voonly` location estimates at each point in time. Note rapid error growth.



(a)



(b)



(c)

Fig. 10. (a,b) `gpsins` location estimates (blue $+$) and `voins` location estimates (green $\square$) in a circular flight path.
(c) Euclidean distance between `gpsins` and `voins` location estimates at each point in time. Notice the slow growth of error as compared to Figure 9(c).

two vectors representing where the point was observed in the image plane ($\vec{x'}$ and $\vec{x}$), and the translation vector between the two camera locations, will all lie on the epipolar plane.[5] By assigning a unique coordinate frame to each camera location, this constraint is represented by the formula

$$\vec{x'}[\vec{p}_{c1}^{c2}]_\times \mathbf{C}_{c1}^{c2}\vec{x} = 0, \quad (47)$$

where $\mathbf{C}_{c1}^{c2}$ is the direction cosine matrix between the camera coordinate frames and $[\vec{p}_{c1}^{c2}]_\times$ is the position of camera 1 in the second camera's coordinate frame, put into a skew-symmetric matrix. (In other words, we calculate the cross product between $\vec{p}_{c1}^{c2}$ and $\mathbf{C}_{c1}^{c2}\vec{x}$.) This constraint enforces that the three vectors $\vec{x'}$, $\mathbf{C}_{c1}^{c2}\vec{x}$, and $\vec{p}_{c1}^{c2}$ all lie within the same plane in the 3-D world.

2) Utilizing the epipolar constraint in a fusion environment: To utilize the epipolar constraint in a fusion environment, we created an Unscented Kalman Filter (UKF) framework. The state in the Kalman filter must contain enough information to generate both $\mathbf{C}_{c1}^{c2}$ and $\vec{p}_{c1}^{c2}$ during the measurement step. To represent general motion by an MAV between two different locations, the UKF state $\vec{X} = \begin{bmatrix} \chi_t \\ \chi_{t-1} \end{bmatrix}$ is used, where χ_t is the navigation state estimate at time t and χ_{t-1} is the navigation state estimate at the previous time. The navigation state at each time contains p_t, the position of the camera at time t in the inertial frame, $\mathbf{C}_t$, the direction cosine matrix relating the inertial frame to the current camera coordinate frame, and v_t, the velocity of the camera at time t. This method for setting up the UKF to enable vision and inertial information fusion was first introduced in [65].

Performing the time update

The time update for this UKF implementation takes two different forms. The first form updates χ_t (the first 10 elements of the current state) every time an IMU measurement occurs. This makes the first 10 elements of the state the most recent navigation state estimate. After each measurement update, the state is also updated using the formula

$$\vec{X}^+ = \mathbf{A}\vec{X}^- \text{ where} \quad (48)$$

$$\mathbf{A} = \begin{bmatrix} I & 0 \\ I & 0 \end{bmatrix}, \quad (49)$$

causing the current state to become $\vec{X} = \begin{bmatrix} \chi_t \\ \chi_t \end{bmatrix}$. The first 10 elements of the state vector are then updated by IMU measurements to realize a new current state $\mathbf{X} = \begin{bmatrix} \chi_{t+1} \\ \chi_t \end{bmatrix}$. With this technique, the two most recent χ estimates, corresponding to the time of the two most recently captured images, are always stored as the current state.

[5]Note that image locations are typically in pixels, while the discussion of vectors so far assumes all vectors are in an unscaled, Euclidean space. In this work, we assume that the camera has been calibrated a-priori and that the conversion from image to Euclidean vectors has already occurred using this calibration information.

Performing the measurement update

To utilize the epipolar constraint as a measurement in a UKF framework, the "Dynamic Vision" approach introduced in [63] is used. Assuming a feature has been detected in two images, the locations of the features are represented by $\vec{x'}$ and $\vec{x}$. Because the epipolar constraint should always be equal to zero, the "measurement" used by the UKF is a vector of zeros in length equal to the number of corresponding features found between the two images. The predicted measurement is $\vec{x'}[\vec{p}_{c1}^{c2}]_\times \mathbf{C}_{c1}^{c2}\vec{x}$ (from Equation (84)) for each set of features $\vec{x'}$ and $\vec{x}$, where $\vec{p}_{c1}^{c2}$ and $\mathbf{C}_{c1}^{c2}$ are functions of χ_t and χ_{t-1}.

While the Dynamic Vision method yields good results in certain cases, it does exhibit some weaknesses that need to be addressed for use on an MAV. First, the translation direction estimates are biased in the direction the camera is pointing. Second, as with all vision-based approaches, it does not estimate the magnitude of translation. We propose methods for overcoming these weaknesses in the following section.

B. Improving the fusion of visual and IMU sensors

In this section, we propose three modifications to baseline epipolar constraint-based fusion of IMU and visual information to significantly increase the accuracy of navigation state estimation on MAVs. These modifications overcome the centering bias, scale ambiguity, and sampling rate problems discussed in the introduction.

1) Overcoming bias toward the center of image points: When using the epipolar constraint to fuse inertial and vision sensors together, the fusion system introduces a bias in estimated translation direction toward the center of the points observed from frame to frame. To understand the source of this error, let us analyze the epipolar constraint when the result of Equation (84) is not zero. The value $x'[\vec{p}_{c1}^{c2}]_\times \mathbf{C}_{c1}^{c2}x$ can be rewritten as a cross product of two vectors followed by a dot product of two vectors. The final results of this computation will be

$$||x'||\ ||\vec{p}_{c1}^{c2}||\ ||x|| \sin(\theta_{\vec{p}_{c1}^{c2} \to \mathbf{C}_{c1}^{c2}x}) \cos(\theta_{x' \to [\vec{p}_{c1}^{c2}]_\times \mathbf{C}_{c1}^{c2}x}), \quad (50)$$

where $\theta_{\vec{p}_{c1}^{c2} \to \mathbf{C}_{c1}^{c2}x}$ is the angle between $\vec{p}_{c1}^{c2}$ and $\mathbf{C}_{c1}^{c2}x$ and $\theta_{x' \to [\vec{p}_{c1}^{c2}]_\times \mathbf{C}_{c1}^{c2}x}$ is the angle between x' and $[\vec{p}_{c1}^{c2}]_\times \mathbf{C}_{c1}^{c2}x$.

As mentioned previously, the correct magnitude for these values is 0. Therefore, the UKF will attempt to set χ_t and χ_{t-1} in its state vector such that the resulting $\vec{p}_{c1}^{c2}$ and $\mathbf{C}_{c1}^{c2}$ minimizes the set of all measurements. However, there are two ways to push the set of measurements toward zero: (1) the epipolar constraint can be met by setting $[\vec{p}_{c1}^{c2}]_\times \mathbf{C}_{c1}^{c2}x$ to be orthogonal to x' (i.e., set $\cos(\theta_{x' \to [\vec{p}_{c1}^{c2}]_\times \mathbf{C}_{c1}^{c2}x}) = 0$), or (2) $\vec{p}_{c1}^{c2}$ can be set parallel to $\mathbf{C}_{c1}^{c2}x$ (i.e., set $\sin(\theta_{\vec{p}_{c1}^{c2} \to \mathbf{C}_{c1}^{c2}x}) = 0$). To meet the first condition, $\mathbf{C}_{c1}^{c2}x$, p_{c1}^{c2}, and x' must all lie in a plane, the original justification behind the epipolar constraint. The second condition, however, can be met by setting the direction of p_{c1}^{c2} equal to x'. Because of this second condition, the results of the epipolar constraint can be pushed to zero

TABLE I

THIS TABLE DEMONSTRATES THE ACCURACY IMPROVEMENTS ACHIEVED BY REMOVING THE sin TERM FROM THE EPIPOLAR CONSTRAINT COMPUTATION. EACH ENTRY LISTS THE MEAN AND STANDARD DEVIATION OF THE ERROR, IN METERS. NOTE THAT THE AVERAGE p_z ERROR HAS DECREASED FROM 758 TO 12 METERS, DEMONSTRATING THE EFFECT OF REMOVING THE BIAS FROM THE EPIPOLAR COMPUTATION.

	p_x error	p_y error	p_z error
With sin (μ, σ)	(-607.0, 136.6)	(2.83, 76.5)	(757.5, 318.1)
sin removed (μ, σ)	(23.1, 85.9)	(-1.77, 16.0)	(11.8, 9.71)

TABLE II

THIS TABLE DEMONSTRATES THE ACCURACY IMPROVEMENTS ACHIEVED BY FUSING THE PITOT TUBE MEASUREMENTS WITH VISION AND IMU INFORMATION. NOTE THAT THE STANDARD DEVIATION OF ERROR ON THE p_x TERM HAS DECREASED FROM 86 TO 2.

	p_x error	p_y error	p_z error
Without pitot tube (μ, σ)	(23.1, 85.9)	(-1.77, 16.0)	(11.8, 9.71)
With pitot tube (μ, σ)	(-0.47, 1.92)	(-1.01, 11.2)	(2.85, 1.08)

by setting $\vec{p}_{c1}^{c2}$ to be as close to parallel to the set of $\mathbf{C}_{c1}^{c2}x$ vectors as possible. Therefore, the translation direction after the UKF measurement update is biased toward the center of the feature points that have been tracked in the second image.

To overcome this biasing of the translation direction, we propose modifying the measurement step of the UKF to eliminate the $\sin(\theta_{\vec{p}_{c1}^{c2}\rightarrow\mathbf{C}_{c1}^{c2}x})$ term from the measurement. To eliminate the effect of $\sin(\theta_{\vec{p}_{c1}^{c2}\rightarrow\mathbf{C}_{c1}^{c2}x})$, the term $[\vec{p}_{c1}^{c2}]_\times\mathbf{C}_{c1}^{c2}x$ is first computed and then normalized to be of length one. The inner product of this term with x' is then taken and returned as the predicted measurement.

To determine the results of this modification, we simulated a 16 second flight of an MAV traveling 200 meters in a straight line. (More details on our simulation environment can be found in Section VII-B.) In Table I, we show the average and standard deviation of the error in the final estimated position of the MAV, both before and after the sin removal modification. Note that before sin removal, the p_z location error mean is a large positive number. This is a result of the bias inherent in the unmodified epipolar fusion environment. After removing the sin as discussed above, the z error is dramatically decreased.

Despite the fact that the p_z error has been decreased by removing the sin term from the epipolar constraint, the p_x error is still quite significant. This error is an artifact of vision-based techniques where there is always a scale ambiguity in the direction of travel (in this case, along the x axis). In the next subsection, we discuss how to reduce the error present in the direction of travel of the MAV due to the scale ambiguity.

2) Removing the scale ambiguity: To remove the large amount of error present in the direction of travel of the MAV, we propose integrating another sensor into the UKF framework discussed above. On a fixed-wing MAV, a pitot tube designed for measuring airspeed can be utilized to measure the current velocity of the MAV. To integrate this measurement in the UKF, we utilize the property discussed in [66] that if measurements are uncorrelated, they can be applied during separate measurement updates of the Kalman Filter. Therefore, whenever the pitot tube is read (approximately 10Hz), the current magnitude of the velocity in the state is computed as a predicted measurement, with the air speed measured by the pitot tube used as a measurement to the UKF.

By applying this measurement at 10 Hz, significant gains in accuracy were achieved. In Table II, we present the results of this modification using the same simulation setup as described for Table I. Note that both the mean and standard deviation of error has decreased in all three location parameters, demonstrating the importance of overcoming the scale ambiguity in visual measurement.

3) Determining the optimal image sampling rate: Typically, when fusing information together, the more information that is available, the more accurate the final result will be. However, with epipolar based visual and inertial fusion, this is not the case. In this subsection, we show that it best to sample the imaging sensors at the *minimal* sampling rate allowed. We also discuss what limits the minimal possible sampling rate.

In Figure 11, we show plots of the mean squared error (MSE) in the estimated final location of the MAV for different image sampling rates. The MSE represents the error in estimated location after a 15 second, straight-line flight. Note that the lowest MSE point does *not* lie at the maximal sampling rate. This can be explained by noting that fusing with the epipolar constraint helps to reduce the amount of error present between two navigation state estimates (i.e., relative error). The total error at the end of flight is going to be the summation of all the relative estimation errors during the flight. Therefore, if the same relative error is achieved by each measurement of the epipolar constraint, but fewer measurements occur, the total error will be reduced. This leads to the counter-intuitive fact that the minimal, as opposed to maximal sampling rate, is ideal for epipolar constraint-based information fusion.

Despite the general rule that the minimal sampling rate is ideal, there are secondary considerations that must be taken into account when deciding on a sampling rate. Note that in Figure 11, the MSE is *not* monotonically decreasing as the sampling rate decreases. There are two principal causes for the increase in MSE at lower sampling rates.

First, the time update of the IMU introduces error into the estimated navigation states, which the UKF attempts to correct using the epipolar constraint. This correction applied by the UKF is a linear correction. As the distance between the estimated and measured navigation states increase however, the linear assumption becomes invalid. Therefore, if too much noise has been added by the IMU, it will not be possible for

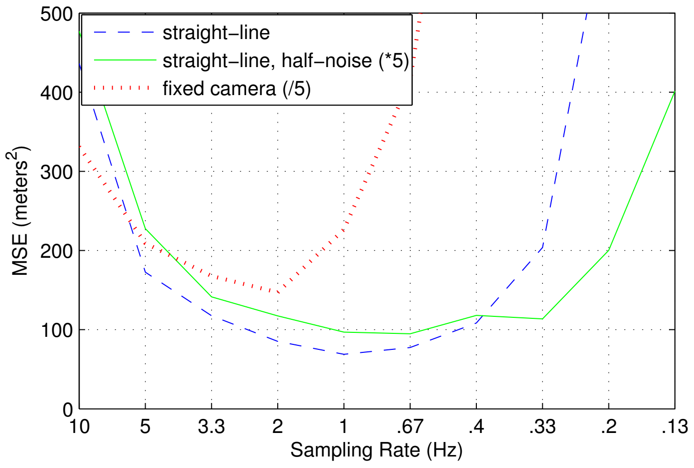


Fig. 11. Results of fusion for different image sampling rates (x-axis) and different fusion setups (different plots). Results are mean squared error (in meters^2). To make all plots appear on the same axes, two of the plots have been scaled by 5 (up and down) as denoted in the legend.

the linear update from the epipolar constraint to correct the IMU-introduced noise. This is demonstrated by the "straight-line" and "straight-line, half-noise" plots in Figure 11. The only difference between the simulation setup of the different plots is that the IMU noise was halved for the "straight-line, half noise" plot. Note that as the IMU noise is decreased, the "optimal" sampling rate becomes lower (moving from 1 Hz to .67Hz), demonstrating that there is a limit placed on the minimal sampling rate by the noise present in the IMU.

Second, the minimal sampling rate is limited by how long the camera can track features in the image. To demonstrate this fact, a simulation was run where, rather than tracking a set of objects throughout the entire flight, as in the "straight-line" simulations of Figure 11, a fixed camera was used so that objects would leave the field of view more quickly. This is the "straight-line, fixed-camera" plot in Figure 11. Note that the sampling rate with the minimal MSE is higher (2Hz) for this plot than the "straight-line" plot (1Hz) where the same world points are observed throughout the MAV flight. Therefore, when applying epipolar constraint-based fusion, it is best to apply the minimal sampling rate that is allowed by (1) the noise present in the IMU and (2) the persistence of features across the images.

C. Estimating inertial sensor biases

Using the modifications proposed in the prior section, it is possible to *overcome* a significant amount of error introduced by inertial sensors when navigating an MAV. In this section, we discuss how fusion based on the epipolar constraint can be used to estimate biases of the inertial sensors, thereby *reducing* the noise from those sensors.

Typical low-cost inertial sensors will have several different *types* of noise [67], [68]. The Kalman Filter setup described in Section IV-A essentially assumes that all noise from the inertial sensors is uncorrelated over time. While the existence of correlated noise (*bias* for the remainder of this section) in the inertial sensors can be overcome by simply increasing the uncorrelated noise covariance estimates, it is preferable to estimate and remove the bias, thereby reducing the amount of noise present in the inertial sensor measurements.

Before describing our method for estimating the bias of the inertial sensors, it is important to demonstrate the feasibility of estimating biases despite the fact that epipolar constraint measurements yield only relative navigation information. In this section, we first demonstrate that it is possible to estimate biases using relative navigation measurements. We then describe our modifications to the Kalman filter framework described in Section IV-A to enable estimation of the inertial sensor biases.

1) Proof of ability to estimate biases: To prove the feasibility of estimating inertial sensor biases, we will perform an observability analysis of a simplified system with a setup that is very similar to our complete MAV navigation system. Our simplified system consists of a state vector with two locations, l_t and l_{t-1}. Similar to the situation where the accelerometers are used to update the current velocity estimates, we use an external rate measurement ($\hat{\dot{l}}_t$) to update the current locations. To estimate the bias on this measurement, we modify the state vector to include the bias, obtaining a state vector of:

$$\vec{x}_t = [l_t \quad l_{t-1} \quad b]^T \,, \tag{51}$$

where b is the bias of the sensor.

The time update for this state over time Δt is

$$\vec{x}_{t+1} = \mathbf{F}\vec{x}_t + \mathbf{G}\hat{\dot{l}}_t \tag{52}$$

where

$$\mathbf{G} = [\Delta t \quad 0 \quad 0] \tag{53}$$

and

$$\mathbf{F} = \begin{bmatrix} 0 & 1 & -\Delta t \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{54}$$

If the measurement of the system provides relative measurements (like vision does for MAV motion), then the observation matrix is:

$$\mathbf{H} = [1 \quad -1 \quad 0]. \tag{55}$$

With this simplified system, we can analyze the observability of b for this Kalman Filter setup. To prove observability, the rank and null vectors of the matrix

$$\mathcal{O} = \begin{bmatrix} \mathbf{H} \\ \mathbf{HF} \\ \mathbf{HFF} \\ \cdots \end{bmatrix} \tag{56}$$

must be found. Substituting in for $\mathbf{H}$ and $\mathbf{F}$ in the first three rows of $\mathcal{O}$, we obtain:

$$\mathcal{O} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 0 & -\Delta t \\ 0 & 0 & -\Delta t \end{bmatrix} \tag{57}$$

By inspection, we find that this system can observe two modes of the system, namely $[1 \quad -1 \quad 0]$ and $[0 \quad 0 \quad 1]$. Because of the second mode, we conclude that biases are observable when relative measurements of the state are used in a Kalman Filter framework.

2) Filter setup for inertial bias estimation: Knowing that biases are observable when relative navigation measurements are used, we can modify the fusion framework developed in Sections IV-A and IV-B to estimate biases. First, we modify the state vector to include biases for all sensors, yielding

$$\vec{X} = [\chi_t, \chi_{t-1}, b_{ax}, b_{ay}, b_{az}, b_{gx}, b_{gy}, b_{gz}]^T, \tag{58}$$

where b_{nm} is the bias estimate with the n subscript denoting a for accelerometer and g for gyro, and the m subscript denoting which axis the sensor is measuring (x, y, or z). The $\mathbf{A}$ matrix described in Equation (86) is modified to

$$\mathbf{A} = \begin{bmatrix} I_{10\times10} & 0 & 0 \\ I_{10\times10} & 0 & 0 \\ 0 & 0 & I_{6\times6} \end{bmatrix}, \tag{59}$$

to maintain bias estimates between measurements from the visual sensor. The time update that utilizes the inertial sensors to update the first 10 elements of the state is modified to subtract out the bias estimate from the inertial measurements. The measurement step for the filter remains unchanged. In the following section, we present results demonstrating the improved navigation performance obtained from estimating the biases of the inertial sensors.

D. Results

To demonstrate the results of fusing visual, air pressure, and inertial sensors together as proposed in this section, we developed a detailed simulator that generated synthetic MAV flights, together with uncorrupted and corrupted sensor measurements for that flight. In the subsections that follow, we first describe the simulation environment in more detail, followed by results demonstrating the efficacy of the fusion system proposed in this section.

1) Simulation Environment: To enable an evaluation of our epipolar constraint-based fusion environment, we first need to generate the true navigation states of the MAV over time. To generate true location data about the MAV, a Bézier curve representing the true path of the MAV was created. A Bézier curve was chosen due to its inherent flexibility in representing many different types of curves in 3-D space. In addition, Bézier curves are a polynomial function of a single scalar t, yielding two significant advantages. First, the location at any time can be easily determined. Second, by differentiating the polynomial with respect to t, the velocity and acceleration at any point on the curve can be computed in closed form. All quantities are assumed to be in a "navigation frame" which has North as its x axis, East as its y axis, and straight down as the z axis. The origin of this frame was arbitrarily chosen as a location on the ground in Utah, near Brigham Young University (close to our MAV flight test area).

In addition to generating the location, velocity, and acceleration of the MAV, we also need to generate the angular orientation (attitude) of the MAV camera. We have used two basic approaches to generating the attitude of the MAV camera. First, for a "fixed" camera, the angular orientation is always constant within the MAV body frame. Second, we set the attitude of the camera such that a specified world location will always be in the center of the image, representing a gimbaled camera that remains pointed at a specific location. We utilize the second approach for the results presented in this section.

Once the true location and attitude of the camera are known, the inputs to the fusion algorithm are generated. We assume the inputs from the IMU consist of 3-axis accelerometer and gyroscope (gyro) readings. To generate accelerometer readings, the acceleration of the camera is computed from the Bézier curve. The effects of gravity, Coriolis, and the rotation of the earth are then added to the accelerometer readings as described in [69], yielding noise-free accelerometer readings. To generate gyro readings, the attitude at two locations on the Bézier curve is computed. The locations on the curve are separated by the gyro sample time. The difference in attitude is then used to compute the angular rates of the camera, yielding noise-free gyro readings. Noise-free pitot tube readings are computed as the magnitude of the velocity at a point on the Bézier curve.

Once the noise-free readings have been computed, two types of noise are added to the sensor readings. First, Gaussian, zero-mean white noise is added to the computed readings. The variance of the noise values were chosen to approximate measurement errors observed on a Kestrel autopilot [3]. Second, a constant bias is added to the gyro and accelerometer readings. For each run of the simulator, biases were randomly selected from a Gaussian distribution with twice the standard deviation of the white noise for that sensor.

To simulate inputs from the camera, a set of random world points to be imaged are created. Using the locations of the world points and the location and attitude of the MAV over time, a set of feature locations corresponding with time along its flight path are created. Features locations for a specific MAV location and attitude are computed using the formula

$$\lambda \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \mathbf{K}\mathbf{C}_n^c(\vec{X}^n - \vec{p}^n), \tag{60}$$

where $\vec{X}^n$ was the location of the world point (in the navigation frame), $\vec{p}^n$ is the position of the camera in navigation frame coordinates (determined from its point on the Bézier curve), $\mathbf{C}_n^c$ is the direction cosine matrix from the navigation frame to the camera frame (also a function of location on the Bézier curve), $\mathbf{K}$ is the calibration matrix of the camera, mapping from Euclidean to pixel locations, λ is a scale factor used for normalizing the third element of the image frame vector to 1, and x_i and y_i are the image coordinates of the point.

After determining the location of the object in the image space, Gaussian white zero-mean noise is added to the image location. We set the standard deviation of the noise equal to a single pixel in the image plane. After adding noise, the pixel values are then "de-calibrated" (multiplied by $\mathbf{K}^{-1}$) to obtain vectors in the same Euclidean space as the MAV navigation state.

2) Fusion Results: To test the efficacy of our proposed fusion environment, we use two different "flight scenarios." In the first scenario, the MAV moves in a straight line starting at 100 meters above the ground and 100 meters south of the navigation frame origin. The camera then moves in a straight line to 100 meters north of the navigation frame origin, holding a constant altitude. In the East-West (y) direction, the MAV is always at 0. Along this path, 161 images were captured at a rate of 10Hz, requiring 16 seconds to fly this path. These values were chosen to achieve an airspeed (12.5 m/s) typical of MAVs. In addition, 1601 samples of the gyro and accelerometer readings were collected. Note that while this flight scenario may seem like an overly simplistic maneuver (flying in a straight line), it was chosen because it actually exacerbates one of the fundamental problem of vision, the universal scale ambiguity. Therefore, this scenario is one of the most difficult scenarios for vision-aided navigation. Results for this scenario are shown in Table III. Note that this scenario was used for the partial results presented earlier in this section.

The second scenario represents a more generic flight of an MAV. It starts at -100 meters north, 100 meters in altitude. It then flies an "S" pattern, going northeast before turning to go northwest. While flying northwest, it passes directly over the navigation frame origin, after which it turns back to head northeast, arriving at -30 meters east, 100 meters north. During the course of the flight, the altitude also drops from 100 meters to 60 meters. This entire flight takes 19 seconds. We refer to this scenario as "The S Pattern", with results shown in Table IV.

In both scenarios described above, the world points being observed were distributed using a three-dimensional Gaussian distribution centered about the navigation frame origin. To keep the objects in view, the camera is continuously rotated to "look at" the origin.

To determine the overall accuracy of each fusion technique, we ran each UKF filter setup with each flight path scenario 100 times. In Tables III, and IV, the mean and standard deviation of the errors across 100 runs of the filter are shown. The mean and standard deviation achieved using only the IMU is also shown for each flight scenario as a reference. The units for the final position error (p_x, p_y, and p_z) are in meters, while the final attitude errors are in degrees. The attitude errors are the amount of yaw (ψ), pitch (θ) and roll (ϕ) that would be required to move from the true location to the estimated locations.

For each of these flight scenarios, five different setups of our UKF environment were used. First, we ran epipolar constraint-based fusion without any of the modifications introduced in Section IV-B (*Baseline*). Second, we remove the bias in the direction the camera is pointed as discussed in Section IV-B1 (sin *Removed*). Third, the measurements from the pitot tube are included in the UKF framework (*Pitot Added*). Fourth, a slower sampling rate (2Hz, as opposed to 10Hz) is used in addition to all the other modifications (*Min. Sampling*). Finally, the "Min. Sampling" filter is modified to estimate the inertial sensor biases (*Est. Bias*).

As shown in these tables, each modification proposed in this section significantly reduces the mean and standard deviation of the error. By including all four proposed modifications, more than an order of magnitude decrease in error is achieved from both IMU-only navigation and the baseline fusion approach. This demonstrates the necessity of including the proposed modifications when considering epipolar constraint based fusion for navigation. It also demonstrates the advantages of estimating the inertial sensor biases to help reduce noise.

V. SIMULATION-BASED COMPARISON

In the prior three section, techniques for navigating using inertial and visual sensor fusion have been introduced. In Section II, a method that assumes GPS is available was introduced, while Sections III and IV are more appropriate for GPS-denied navigation. The two methods introduced in Sections III and IV, however, are both based on frame-to-frame measurements of image features. This approach has significant advantages in terms of computational feasibility, utilizing existing methods for outlier rejection, and minimizing the long-term effects of non-linearities on the underlying estimation routine.

Despite the advantages of two-frame methods, significant research work has also been performed where the location and attitude of the camera and the location of objects observed by the camera are estimated simultaneously, leading to the Simultaneous Localization And Mapping (SLAM) problem. In this section, we seek to improve our understanding of the benefits incurred from the increased cost in complexity for SLAM-based methods compared to frame-to-frame (constraint-based) methods. Note that there has been a significant amount of work describing the theoretical and practical limitations of SLAM-based algorithms [76], [77], [78]. As far as we know, however, this is the first work which directly compares these two fundamentally distinct approaches to visual/IMU sensor fusion.

TABLE III
MEAN AND STANDARD DEVIATION OF ERROR IN THE FINAL ESTIMATED LOCATION AND ATTITUDE OF THE MAV WHEN FLYING A STRAIGHT-LINE PATH WITH A GIMBALED CAMERA. NOTE THAT THE ERRORS IN ATTITUDE ARE ALL UNDER ONE DEGREE WHEN ALL FOUR MODIFICATIONS PROPOSED IN THIS SECTION ARE IMPLEMENTED.

	IMU only	**Baseline**	sin **Removed**	**Pitot Added**	**Min. Sampling**	**Est. Bias**
p_x **Error** (μ,σ)	(-4.48, 115.4)	(-607.0, 136.6)	(23.1, 85.9)	(-0.47, 1.92)	(-0.39, 1.19)	(0.03, 1.13)
p_y **Error** (μ,σ)	(8.27, 95.3)	(2.83, 76.5)	(-1.77, 16.0)	(-1.01, 11.2)	(-0.24, 3.05)	(-.08, 1.96)
p_z **Error** (μ,σ)	(15.1, 12.7)	(757.5, 318.1)	(11.8, 9.71)	(2.85, 1.08)	(11.1, 1.17)	(2.19, 0.82)
ψ **Error** (μ,σ)	(-0.69, 14.0)	(2.26, 17.1)	(0.20, 2.52)	(0.21, 2.71)	(0.09, 1.07)	(0.02, 0.58)
θ **Error** (μ,σ)	(0.16, 15.93)	(-143.5, 61.3)	(1.57, 6.59)	(-0.31, 1.70)	(1.23, 1.78)	(0.27, 0.61)
ϕ **Error** (μ,σ)	(-1.05, 12.9)	(2.90, 18.1)	(0.65, 5.67)	(0.26, 4.13)	(0.08, 1.13)	(0.02, 0.72)

TABLE IV
MEAN AND STANDARD DEVIATION OF ERROR IN THE FINAL ESTIMATED LOCATION AND ATTITUDE OF THE MAV WHEN FLYING THE "S-CURVE" PATH WITH A GIMBALED CAMERA.

	IMU only	**Baseline**	sin **Removed**	**Pitot Added**	**Min. Sampling**	**Est. Bias**
p_x **Error** (μ,σ)	(0.26, 110.4)	(-128.6, 130.9)	(-22.8, 88.1)	(-1.89, 23.1)	(0.79, 2.39)	(0.08, 1.73)
p_y **Error** (μ,σ)	(-15.4, 132.4)	(78.0, 354.0)	(-11.4, 92.7)	(0.91, 11.8)	(6.20, 2.75)	(2.09, 1.82)
p_z **Error** (μ,σ)	(18.8, 33.6)	(528.3, 278.9)	(41.4, 102.0)	(-2.90, 4.04)	(5.18, 2.79)	(-1.58, 1.36)
ψ **Error** (μ,σ)	(0.91, 14.5)	(39.3, 31.4)	(5.01, 19.4)	(0.38, 3.20)	(-0.23, 2.67)	(0.26, 1.40)
θ **Error** (μ,σ)	(-0.07, 11.5)	(-27.1, 26.9)	(-3.84, 23.0)	(-1.18, 4.91)	(0.78, 1.87)	(-0.41, 0.99)
ϕ **Error** (μ,σ)	(2.32, 13.2)	(-55.7, 47.9)	(-1.40, 25.6)	(1.85, 5.59)	(0.71, 1.28)	(0.65, 0.73)

While a significant amount of research has been published in both SLAM-based and constraint-based IMU/vision fusion (including works demonstrating their efficacy with real video and IMU information), there are some significant difficulties in comparing these results. Some of the difficulties in comparing results include:

- How to identify a feature across multiple images (the correspondence or data association problem) differs greatly from implementation to implementation. The results of any algorithm are greatly dependent on how well features are associated over time, making direct comparison of published results difficult.
- Several different options exist for the "fusion engine" used in these approached. Possible options include: (1) the Extended Kalman Filter [79], (2) the Unscented Kalman Filter [80], (3) the Information Filter [79], or (4) the Particle Filter [81]. The exact methodology used to implement each of these engines can also differ greatly from published work to published work.
- The specific camera and IMU setup used to get final results differs greatly between implementations.
- What noise is present in each of the sensors can vary greatly from implementation to implementation, yielding significantly different final results.
- When fusing data from multiple sensors, the methods used to synchronize their timing and precisely align their geometric orientations with respect to each other can cause significantly different results to be reported from the same basic fusion techniques.

Because of these and several other small differences in implementations between published works, it is impossible to perform a direct comparison between previously published fusion techniques.

To perform a direct comparison between fusion techniques, we assumed three basic similarities between the environments for the techniques : (1) perfect data association across images, (2) an Unscented Kalman Filter (UKF) framework [80] is used as the basic filtering technique in both approaches and (3) the same noise is injected into the system with both scenarios. With these ambiguities removed from the environment, we perform a comparison between navigation *approaches*, rather than navigation *implementations*.

The remainder of this section is organized as follows. Section V-A briefly reviews the constraint-based approach that we implemented, with a discussion of the novel modifications made to the algorithm to enable a fair comparison to SLAM techniques. Section VI-C describes our SLAM-based approach to navigation for MAVs. Section V-C presents our simulation-based comparison between different navigation approaches.

A. *Constraint-Based Fusion*

The constraint-based method for fusing visual and IMU data together is to use the "epipolar" constraint between two images. The approach used in this section is the same basic approach introduced in Section IV-A. In addition, we used the modification proposed in Section IV-B1 to overcome the bias towards the center of the image. The remaining modifications in Section IV-B, however, were not utilized as they would not enable a fair comparison between constraint-based techniques and SLAM-based techniques. Because we could not assume an external air-speed sensor, another technique had to be utilized to overcome the unobservability of translation magnitude in constraint-based techniques.

Unfortunately, this unobservability of velocity magnitude leads to a positive bias in velocity magnitude, in the true direction of velocity. To understand this effect, consider Figure 12 where a two-dimensional example of how the Kalman filter will merge together translation directions is shown. Due to the scale ambiguity, the measurement step can only correct the direction of translation, not the magnitude. However, when a derivative is taken of the translation direction, it forms a tangent plane that will always lead to an increased magnitude of the resulting vector. Because the increase in translation leads to an increase in velocity, and the IMU only measures accelerations, these increases lead to an exponential increase in error *in the true direction of travel.*

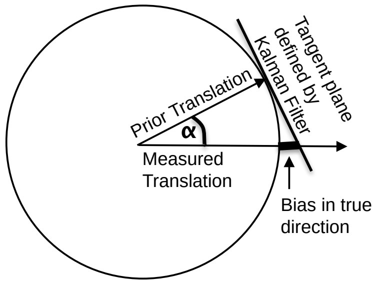


Fig. 12. A simple example of why the translation magnitude is biased in the direction of true translation when performing 2-frame fusion. The measurement step of the Kalman Filter combines the prior estimated direction with the measured direction. However, the derivative of measured angle difference is a tangent plane, leading to increased magnitude in the true direction.

To overcome this weakness in the epipolar constraint fusion method, we used a Kalman-filter type update on the magnitude of velocity after each measurement step. To determine the "update" that should be applied, a discussion of what errors in the velocity may occur is required. Errors in velocity can be broken into two components, errors in the direction of the true velocity, and errors in orthogonal directions (see Figure 12(c)). After the measurement step of the UKF is completed, we assume that the *direction* of the translation is correct. Therefore, we would like to set the magnitude of the post-measurement translation equal to the magnitude of the pre-measurement magnitude *in the direction of the post-measurement translation.*

A simplistic approach to modifying the velocity would be to simply change the velocity after each measurement step (v^+) to the velocity magnitude that was present before the measurement step (v^-). However, simply changing the velocity magnitude will not reverse errors introduced in the accelerometer bias estimates due to coupling between the velocity estimate and the biases. Therefore, a "virtual measurement" is used to correct the change in velocity magnitude and its effects on IMU bias estimates. This virtual measurement, Δv, is found as:

$$\Delta v = (k-1) * v^+, \text{ where} \tag{61}$$

$$k = \frac{v^{-T} v^+}{v^{+T} v^+}, \tag{62}$$

which sets the magnitude of v^+ equal to the magnitude of v^- projected in the direction of v^+. This virtual measurement is used as a quasi-Kalman measurement update as:

$$\begin{aligned} \vec{X}^+ &= \vec{X}^- + K\Delta v \text{ where} \\ K &= PH^T(HPH^T)^{-1}, \\ H &= [0 \quad I_3 \quad 0]. \end{aligned} \tag{63}$$

Note that the covariance matrix is not modified in this step as no real measurement has occurred. Using this method, the measurement step of the Kalman filter does not significantly alter the velocity *magnitude* nor the accelerometer biases in the velocity direction.

B. SLAM-based Fusion

The second type of fusion technique that we evaluated was based on previous Simultaneous Localization and Mapping (SLAM) work. The general concept of SLAM is that a set of world points should be observed (mapped) and localized over time while simultaneously determining the current location of the sensor (localization). By maintaining a full covariance matrix between the location of the sensor and the location of the world points, the SLAM algorithm has been shown to be convergent in specific circumstances [83]. Therefore, SLAM has been an area of intensive research for the past several years. Most SLAM methods, however, have concentrated on utilizing sonar or laser-based sensor for measurements of the world, as opposed to a camera. The subset of SLAM work that has focused on using a camera is generally called bearing-only or visual SLAM. In general, however, these methods have assumed that a camera is the only input, and have not attempted to fuse in IMU data while performing SLAM (although there have been exceptions: [84], [60]).

For our implementation of SLAM, a UKF was once again used, helping make the comparison between SLAM-based and constraint-based techniques equitable. The state of our UKF includes the current navigation state of the camera (p, q, v), the biases of the IMU (b), followed by entries for the world points that are being tracked. One of the primary difficulties with visual SLAM is the initialization of new features. Because only the bearing of a world object from the camera is observed the first time, the 3-D location of the world point cannot be added directly to UKF state for two reasons. First, there is essentially infinite variance in one direction if only one observation has occurred (see Figure 18). Second, because there may be error in the observed bearing, the variance in the directions orthogonal to the infinite variance direction grow with distance from the camera, causing the initial observation of the point to have an extremely non-linear variance.

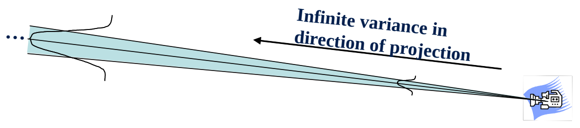


Fig. 13. A simple figure demonstrating the extremely non-linear variance results from a single camera observation of a point. First, there is infinite variance in the direction of the observation. Second, as the distance from the camera increases, the variance in orthogonal directions also increase

Several attempts to overcome this initialization problem have been utilized previously. In [73], a stereo camera system was used, which eliminates the infinite variance problem for world features that are "close enough" to the stereo camera pair. Because the small size of MAVs severely limits the baseline that can be placed between stereo cameras, however, this many not be a feasible option. Therefore, [59] initialized world features observed from an aircraft by assuming points were on the ground and intersecting the ray from the camera with a model of the ground elevation. However, with an MAV that may be used to fly through urban terrain, the assumption of points on the ground may not be valid. In [25], targets of known size were placed on the ground, and the size of the target was used to initialize the depth of the target from the camera. This method is not applicable to our work as we assume world points that are not known a-priori.

To provide an initialization methodology that can be used with a single camera, and does not pre-suppose information about the target, we use the method introduced in [85] to initialize feature locations. This method, the "inverse depth" method, places three new items in the Kalman Filter state for each new object that is observed, namely: (1) the projective center of the camera when the feature was first observed (represented by P_c), (2) the direction of the ray (in 3-space) of the first observation (represented by two angles, θ and ϕ), and (3) the inverse depth of the point along that line (represented by ξ). The inverse depth, rather than the depth, is used because a finite value between 0 and 1 covers the entire depth from 1 to infinity in a linear fashion. Therefore, a variance of .5 on the inverse depth is equivalent to an infinite variance when using depth. In addition, because the bearing of the original observation is included in the Kalman Filter state, the "increasing variance with distance" problem discussed above is properly represented in the Kalman Filter.

While the inverse depth methodology has many advantages, it has a significant failing when it comes to MAV video. In [85], the direction of the ray that first observed the point to be localized is represented by two angles, θ and ϕ. This works very well for the situation outlined in [85], where the camera is generally facing forward. However, in an MAV where the camera may be facing down, this representation has a significant shortcoming. Assume two rays, both pointing almost straight down, but with one facing slightly forward and one slightly to the side. Using the two angles ϕ and θ to represent these locations, the rays will be very far apart (i.e., $\phi = 90°$, $\theta = 0°$ and $\theta = 90°$) Therefore, there is a singularity in the two angle representation around the downward direction. To remove this singularity, we replaced the two-angle representation with a 3-vector (r) that is normalized to 1. To keep the vector normalized during the measurement step, we used the approach outlined in [82].

With the state vector described, we can now describe in detail the time and measurement updates utilized in our SLAM-based fusion technique. During the time update, only the current navigation state estimate of the camera needs to be updated. The updates occur according to the accelerometer and gyro measurements from the IMU. During the measurement step of the UKF, the location of each world point that is currently being observed in the image is used. The predicted measurement for each world location is

$$\vec{x} = \lambda \mathbf{C}_n^c (\vec{P}_c^n + \frac{1}{\xi}\vec{r} - \vec{p}_c^n), \tag{64}$$

where $\mathbf{C}_n^c, \vec{P}_c^n, \xi, \vec{r}$, and $\vec{p}_c^n$ are all derived from the current state estimate.

C. Comparison of Methods

To compare the constraint and SLAM-based imaging and inertial fusion systems, we created a simulation environment that simulates both an IMU and camera for use in the imaging/inertial fusion algorithms. This simulator is described in the following sub-section. The results obtained from adding noise and using each fusion method are described in the subsection V-C2. A discussion of these results concludes this section.

1) Simulation Environment: To generate true navigation states of the MAV over time, a Bézier curve representing the true path of the MAV was created. A Bézier curve was chosen due to its inherent flexibility in representing many different types of curves in 3-D space. In addition, Bézier curves are a polynomial function of a single scalar t, yielding two significant advantages. First, the location at any time can be easily determined. Second, by differentiating the polynomial with respect to t, the velocity and acceleration at any point on the curve can be computed in closed form. All quantities are assumed to be in a "navigation frame" which has North as its x axis, East as its y axis, and straight down as the z axis. The origin of this frame was arbitrarily chosen as a location on the ground in Utah, near Brigham Young University (close to our MAV flight test area).

In addition to generating the location, velocity, and acceleration of the MAV, we also need to generate the angular orientation (attitude) of the MAV camera. We have used two basic approaches to generating the attitude of the MAV camera: (1) for a "fixed" camera, the angular orientation is always constant within the MAV body frame; (2) for a "gimballed" camera, the attitude of the camera is set so that the origin of the navigation frame would be imaged in the center of the image.

Once the true location and attitude of the camera are known, the inputs to the fusion algorithm are generated.

We assume the inputs from the IMU consist of 3-axis accelerometer and gyroscope (gyro) readings. To generate accelerometer readings, the acceleration of the camera is computed from the Bézier curve. The effects of gravity, Coriolis, and the rotation of the earth are then added to the accelerometer readings as described in [69], yielding noise-free accelerometer readings. To generate gyro readings, the attitude at two locations on the Bézier curve is computed. The locations on the curve are separated by the gyro sample time. The difference in attitude is then used to compute the angular rates of the camera, yielding noise-free gyro readings.

Once the noise-free readings have been computed, two types of noise are added to the sensor readings. First, Gaussian, zero-mean white noise is added to the computed readings. The variance of the noise values were chosen to approximate measurement errors observed on a Kestrel autopilot [3]. Second, a constant bias is added to the gyro and accelerometer readings. For each run of the simulator, biases were randomly selected from a Gaussian distribution with 10x the standard deviation of the white noise for that sensor.

To simulate inputs from the camera, a set of random world points to be imaged are created. These points are randomly distributed both in x-y location and about the zero altitude plane (i.e., the points are non-planar.) Using the locations of the world points and the location and attitude of the MAV over time, a set of feature locations corresponding with time along its flight path are created. Features locations for a specific MAV location and attitude are computed using the formula

$$\lambda \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \mathbf{K}\mathbf{C}_n^c(\vec{X}^n - \vec{p}^n), \tag{65}$$

where $\vec{X}^n$ was the location of the world point (in the navigation frame), $\vec{p}^n$ is the position of the camera in navigation frame coordinates (determined from its point on the Bézier curve), $\mathbf{C}_n^c$ is the direction cosine matrix from the navigation frame to the camera frame (also a function of location on the Bézier curve), $\mathbf{K}$ is the calibration matrix of the camera, mapping from Euclidean to pixel locations, λ is a scale factor used for normalizing the third element of the image frame vector to 1, and x_i and y_i are the image coordinates of the point.

After determining the location of the object in the image space, Gaussian white zero-mean noise is added to the image location. We set the standard deviation of the noise equal to a single pixel in the image plane. After adding noise, the pixel values are then "de-calibrated" (multiplied by $\mathbf{K}^{-1}$) to obtain vectors in the same Euclidean space as the MAV navigation state.

2) Setup and Results of Comparison: Using the general setups described above for constraint-based and SLAM-based fusion of the IMU and visual information, we designed three different "flight scenarios" to test the relative merits of each fusion approach. In the first scenario, the camera moves in a straight line starting at 100 meters above the ground, 100 meters south of the navigation frame origin. The camera then moves in a straight line to 100 meters north of the navigation frame origin, holding a constant altitude. In the East-West (y) direction, the MAV is always at 0. Along this path, 151 images were captured at a rate of 10Hz, requiring 15 seconds to fly this path. These values were chosen to achieve an airspeed (13.3 m/s) typical of MAVs. In addition, 1501 samples of the gyro and accelerometer readings were collected. Note that while this flight scenario may seem like an overly simplistic maneuver (flying in a straight line), it was chosen because it actually exacerbates one of the fundamental problem of vision, the universal scale ambiguity. Therefore, this scenario is one of the most difficult scenarios for vision-aided navigation. We refer to this scenario as the "Straight-line Flight" scenario. Results for this scenario, with a gimballed camera, are shown in Figures 14 and 15.

To determine the overall accuracy of each technique, we ran each UKF filter setup 100 times. In Figures 14, 15, 16, and 17, we use the "boxplot" command in MATLAB to display the median and spread of the errors present in the final location and attitude estimate across the 100 runs. In these plots, the line in the middle of the box represents the median of the data, the box represents the middle 50% of the data, and the crosses outside the lines represent outlier points.

(Note that in Figure 14, it is apparent that both the constraint and SLAM-based fusion techniques achieve significant improvements over IMU-only fusion. Similar results were seen with all scenarios tested, so all other figures simply show comparative results between fusion techniques, allowing a more detailed display of results. Figure 15 is the same as Figure 14 except that IMU results have been removed.)

The second scenario tested represents a more generic flight of an MAV. It starts at -100 meters north, 100 meters in altitude. It then flies an "S" pattern, going east before turning back, passing directly over the navigation frame origin, headed in a northwest direction, and eventually turning back to arrive at -30 meters east, 100 meters north. During the course of the flight, the altitude drops from 100 meters to 60 meters. This entire flight takes 18 seconds. We refer to this scenario as "The S Pattern", with results shown in Figure 16.

In both scenarios described above, the world points being observed were distributed using a three-dimensional Gaussian distribution centered about the navigation frame origin. To keep the objects in view, the camera is continuously rotated to be "looking at" the origin, with the "top" of the camera facing in the direction of travel at the current point.

The final scenario used was to determine the response of the different fusion techniques to having a feature in view for only a short period of time. The flight pattern in this case was the same as the "straight-line flight" described above. However, rather than distributing the world points about the navigation frame origin, and always pointing the camera at the origin, 30 points were uniformly distributed in the North-South direction from 150 meters north to 110 meters south. The camera was kept at a fixed attitude of 80 degrees down. In the East-West and down-up direction, the points were distributed using a Gaussian distribution with a standard deviation of 20 meters. This scenarios is referred to as the "straight-line, fixed camera" scenario, and results are shown in Figure 17.

For each of these flight scenarios, three different setups of our UKF environment were used. The most basic comparison of constraint-based and SLAM-based fusion entails running two different UKFs, with the exact same IMU and feature location inputs. We refer to this as the "Constraint1" and "SLAM" setups. For the straight-line flight and S pattern, 10 world points were used.

In addition to comparing the constraint and SLAM-based techniques using the exact same inputs, we also created a case that represents one of the fundamental differences between the two navigation approaches. One of the principal shortcomings of SLAM-based techniques is that, due to the inclusion of world points in the filter state, real-time SLAM filters are intrinsically limited in the number of features they can simultaneously localize. In addition, it is far more difficult to reliably track features over a long time period as required by SLAM as opposed to simply tracking features from one frame to the next. Therefore, it is easier to reliably track a large number of features using the constraint-based technique then SLAM-based techniques. To represent this effect, we also simulated a constraint-based fusion technique (*Constraint2*) that tracks 4 times as many features as the first two setups (i.e., 40 world points for the *straight-line* and *S-pattern* results, and 120 world points for the *straight-line, fixed-camera* results.)

To tune the UKF, we first set the process noise matrix, Q, to exactly equal the sources of noise on the IMU. The Q matrix was then increased to compensate for non-linearities in the process model. The values of Q were increased until the performance of the filter no longer increased, thereby tuning the Q matrix for use in these simulations.

3) Discussion of Results: From the data presented in Figures 14-17, we can draw several conclusions about the relative strengths and weaknesses of the constraint and SLAM-based techniques implemented in this section for vision-assisted navigation. First, let us consider the case when the camera continuously tracks the same objects, whether flying a straight line (Figure 15) or an S-pattern (Figure 16). In both these cases, the average error in the final navigation state estimates for SLAM-based techniques are at least an order of magnitude better than either constraint-based technique. From these results, we conclude that if the MAV is flying in a pattern where the same general area is being observed over time (i.e., in an orbit pattern, perimeter surveillance, etc.), then SLAM-based navigation algorithms are required for accurate navigation.

However, when the camera is fixed and the MAV camera is not continuously observing the same world points (Figure 17), we observe a different pattern in the results. While the SLAM-based fusion technique is better than the *Constraint1* and *Constraint2* cases in position estimation, the *Constraint2* and *SLAM* results are much more comparable. Therefore, when an area is quickly being observed by an MAV and will not be re-observed, then the constraint-based techniques may be useful in MAV flights.

In addition to the observation of what type of flight pattern favors SLAM-based techniques vs. constraint-based techniques, our results also demonstrate some interesting attributes of each fusion technique. Note that in both straight-line flight scenarios (Figures 15 and 17), the standard deviation of the SLAM-based technique for T_x is significantly smaller than the constraint-based techniques. This demonstrates the significance of the global scale ambiguity in vision-based methods. With vision, there is always a scale ambiguity, but the SLAM-based technique attempts to find one scale ambiguity across the entire flight. Constraint-based techniques, on the other hand, have a scale ambiguity at every step of the fusion filter. Therefore, the standard deviation in the direction of travel is much larger using constraint-based techniques than SLAM-based techniques.

VI. Analytical comparison

Several methods have been introduced in the navigation literature to fuse camera and IMU information together. Despite impressive individual results for each method, it is difficult to generalize which techniques are most appropriate and understand precisely why some methods achieve more accurate navigation estimates than others. This is primarily due to the differences between papers in the IMUs utilized, the filter used, and numerous other implementation differences that make a direct, comparative study of pre-existing literature difficult. While the previous section presented a simulation-based comparison of different navigation algorithms, in this section we propose an analytical tool, "drift-order" characterization of navigation algorithms, that can be used to compare different navigation algorithms. This drift-order characterization is similar to the "big O" notation used to characterize the computational complexity "order" of general algorithms.

To characterize the order of an algorithm, the growth in computation time is determined as the number of inputs to the algorithm increases. As the number of inputs becomes large enough, the terms causing the most rapid growth determine the order of the algorithm. Similarly, we propose to evaluate navigation algorithms by the error growth (drift) in their results as time increases, concentrating on the terms with the quickest growth over time. We term this "drift order" notation and will use a "big D" ($\mathcal{D}$) to represent drift order results. With this drift characterization tool, we are able to analytically analyze different camera and IMU fusion-based navigation algorithms. Specifically, we focus on two commonly used approaches to visual and inertial information

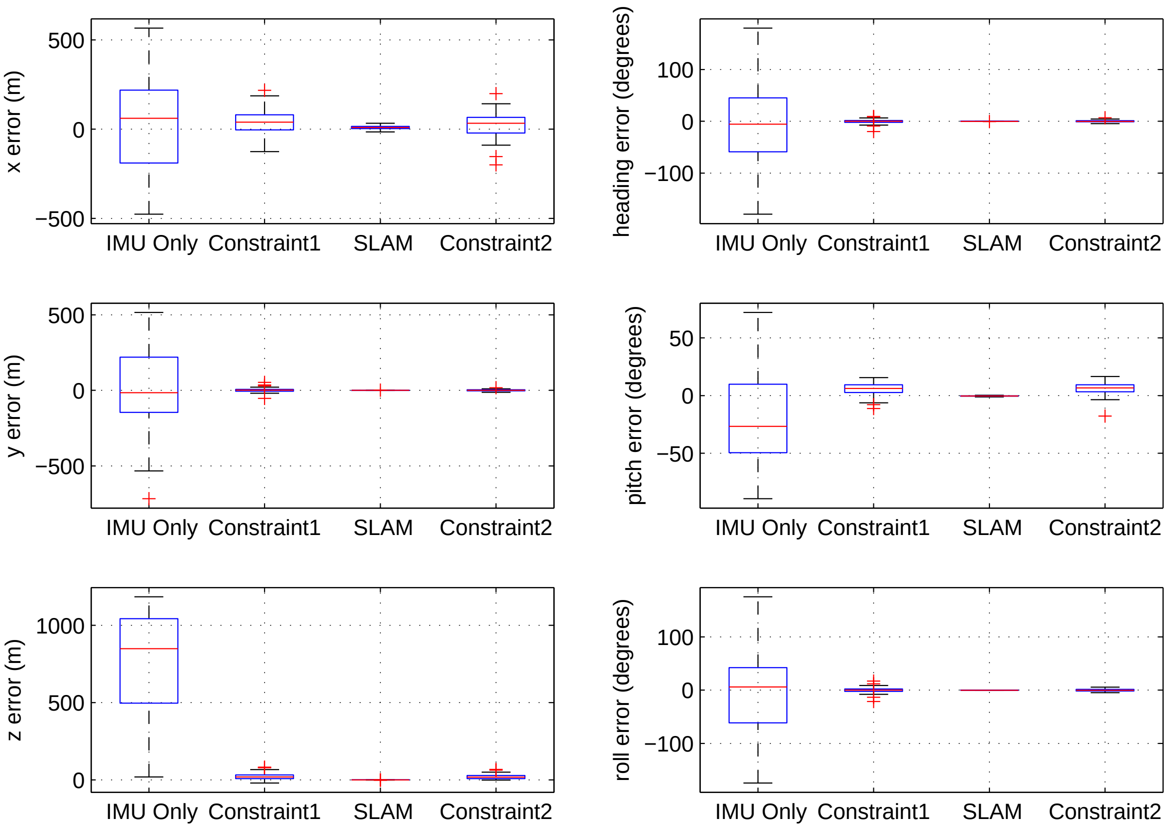


Fig. 14. Comparative results between fusion techniques and IMU-only – the straight line flight

fusion: (1) methods utilizing the epipolar constraint (epipolar-based techniques) and (2) methods solving the simultaneous localization and mapping (SLAM) problem.

4) Related Work: Much of the methodology used in this section to characterize the performance of a navigation system was first introduced in [86]. [86] attempts to answer the question of how quickly the navigation system for a tractor with a radar will drift when GPS signals are lost. This section extends the work in [86] in two primary ways: (1) we use the methodology of [86] to analyze visual and inertial fusion systems, as opposed to a radar-based system and (2) we introduce a formal classification approach (the big D notation) for allowing comparisons between navigation approaches.

One of the techniques analyzed in this section is simultaneous localization and mapping (SLAM) based approaches for fusing visual and inertial information together. Several works have previously attempted to characterize the performance of SLAM algorithms as time goes to infinity[77], [78]. However, each of these approaches assume that a fixed set of features are observed by the SLAM-filter over time, limiting the applicability of the analysis to robots that stay in a small, local area. In this section, we extend that analysis to include feature replacement over time, enabling the analysis of SLAM-based navigation approaches over a much larger area.

It should also be mentioned that there are numerous papers that deal with navigating with an IMU only, using the epipolar constraint for visual and inertial fusion, and using SLAM to fuse inertial and visual information together. A survey of these approaches is beyond the scope of this report, though some references are included with the descriptions of these algorithms in the remainder of this paper. The majority of these paper introduce a novel modification to the basic navigation approach and demonstrate improved results using their method. Note, however, that the results presented in each of these papers depend heavily on the noise models used for the IMU, which IMU was used, which filtering method was used to enable fusion, and numerous other implementation decisions. Therefore, it is not feasible to perform direct comparisons between distinct navigation approaches by simply comparing previously published papers. In this section, we attempt to compare navigation approaches using drift order analysis because the drift order is independent

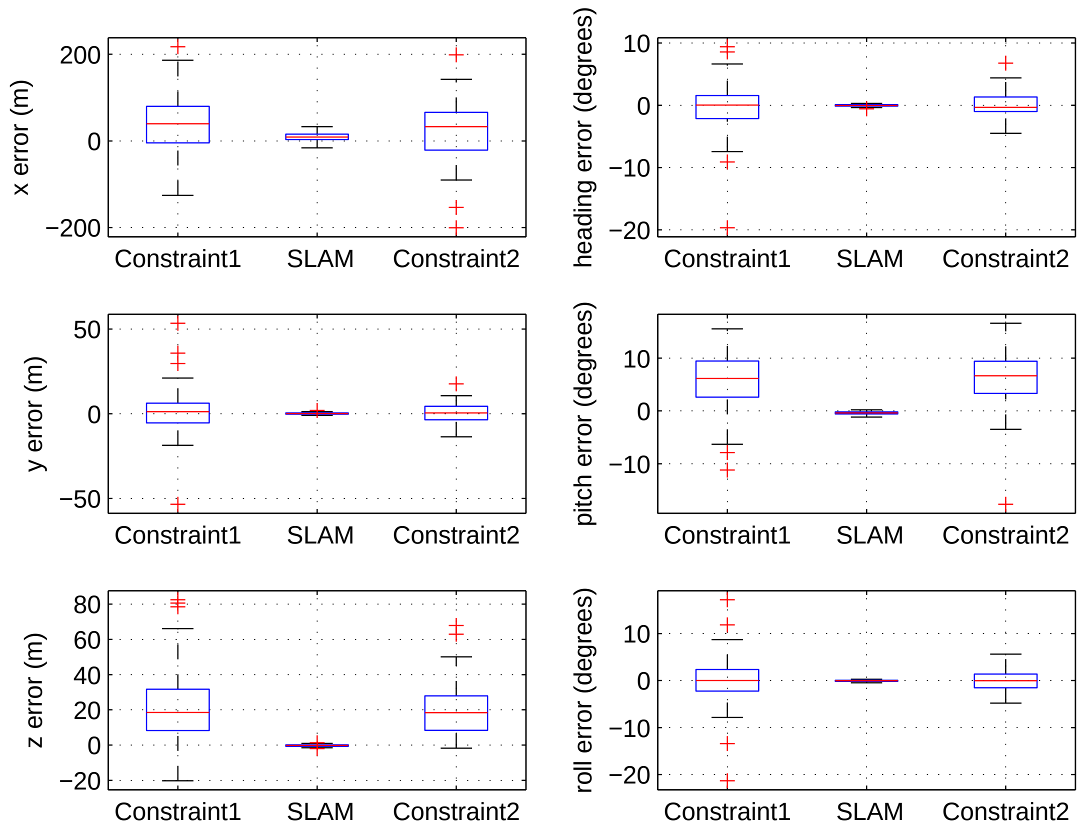


Fig. 15. Comparative results between fusion techniques – the straight line flight

of many implementation issues that differentiate previously published navigation approaches.

The next three sub-sections present a drift order analysis of three navigation approaches: IMU-only (Section VI-A), epipolar-based fusion of visual and inertial data (Section VI-B), and SLAM-based fusion (Section VI-C). In Section VII-B, we present simulation results verifying the drift order results presented in the prior sections.

A. Basic IMU-based Navigation System Performance

To analyze the performance of IMU-only navigation systems, we utilize a simple model of an object that can be completely measured by a single accelerometer and gyroscope. The current navigation state of the object is

$$\vec{x} = \begin{bmatrix} x \\ y \\ \theta \\ V \end{bmatrix}, \tag{66}$$

where x and y represent the 2-D location of the object being navigated, θ is the angular orientation of the object and V is the magnitude of the current velocity. The system is characterized by the following dynamic equations:

$$\begin{aligned} \dot{x} &= V\cos\theta && (67)\\ \dot{y} &= V\sin\theta && (68)\\ \dot{\theta} &= \omega && (69)\\ \dot{V} &= a, && (70) \end{aligned}$$

where ω is angular velocity and a is acceleration of the object. ω is measured using a gyroscope and a is measured using an accelerometer. We assume that each IMU sensor has two sources of noise: (1) white, zero-mean Gaussian noise characterized by its variance σ_w^2 and (2) a constant bias b which was chosen from a Gaussian distribution with variance σ_b^2. Therefore, we have

$$\begin{aligned} \hat{\omega} &= \omega + \mathcal{N}(0, \sigma_{w,g}^2) + b_g && (71)\\ \hat{a} &= a + \mathcal{N}(0, \sigma_{w,a}^2) + b_a, && (72) \end{aligned}$$

where $\mathcal{N}(0, \sigma^2)$ denotes a white, Gaussian noise process with variance σ^2 and zero mean, the g and a subscripts denote gyroscope and accelerometer-related quantities, respectively, and $\hat{\omega}$ represents a measured estimate of ω. Similar notation is used for $\hat{a}$.

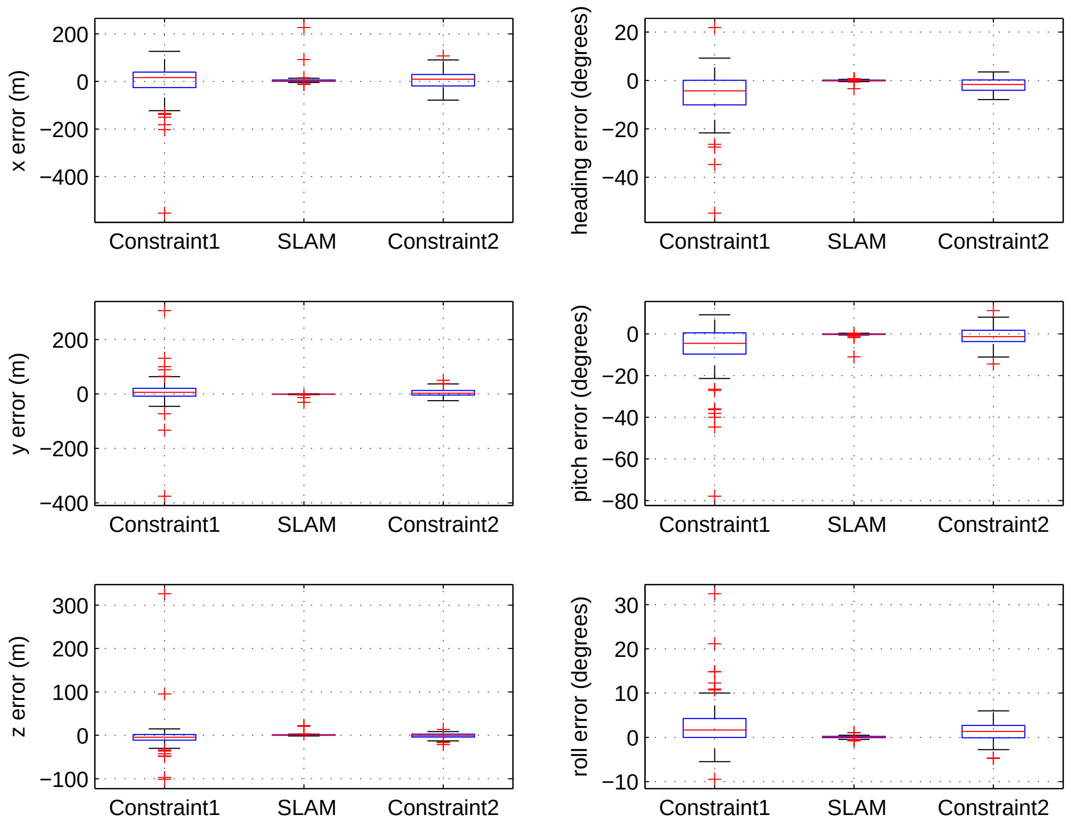


Fig. 16. Comparative results between fusion techniques – the S-pattern

1) Integration of Single Sensor: The first step in analyzing the performance of an IMU-based system is to determine how the noise in the IMU sensors affects estimates which are an integration of the IMU sensors. This includes $\hat{\theta}$, which is an integration of $\hat{\omega}$, and $\hat{V}$ which is an integration of $\hat{a}$. For the remainder of this section, we will concentrate on integrating $\hat{\omega}$ to obtain $\hat{\theta}$, though identical results could be obtained by solving for $\hat{V}$.

To simplify the analysis of integration effects, we use Euler integration, leading to

$$\begin{aligned}\hat{\theta} &= T_s \sum_{i=1}^{k} \hat{\omega} \\ &= T_s \sum_{i=1}^{k} \omega + \mathcal{N}(0, \sigma_{w,g}^2) + b_g \qquad (73) \\ &= \theta + T_s \sum_{i=1}^{k} \mathcal{N}(0, \sigma_{w,g}^2) + b_g,\end{aligned}$$

where T_s is the sampling time of the sensor and k is the number of samples that have been measures so far, leading to $t = kT_s$. To characterize the effect of noise in $\hat{\theta}$, we introduce the term $\tilde{\theta}$ which denotes the error in $\hat{\theta}$. Note that $\tilde{\theta} = \hat{\theta} - \theta$, leading to

$$\tilde{\theta} = T_s \sum_{i=1}^{k} \mathcal{N}(0, \sigma_{w,g}^2) + b_g. \qquad (74)$$

To determine the mean squared error, we compute

$$\begin{aligned}MSE(\tilde{\theta}) &= T_s^2 E\left[(\sum_{i=1}^{k} \mathcal{N}(0, \sigma_{w,g}^2) + b_g)^2 \right] \\ &= T_s^2 E\Bigg[\sum_{i=1}^{k} \sum_{j=1}^{k} \left(\mathcal{N}_i(0, \sigma_{w,g}^2) + b_{g,i}\right) \qquad (75) \\ &\qquad \left(\mathcal{N}_j(0, \sigma_{w,g}^2) + b_{g,j}\right) \Bigg].\end{aligned}$$

Using the properties that (1) the term $\mathcal{N}(0, \sigma_{w,g}^2)$ is uncorrelated across i and j, (2) the bias and white noise are uncorrelated with each other, and (3) the bias is a constant

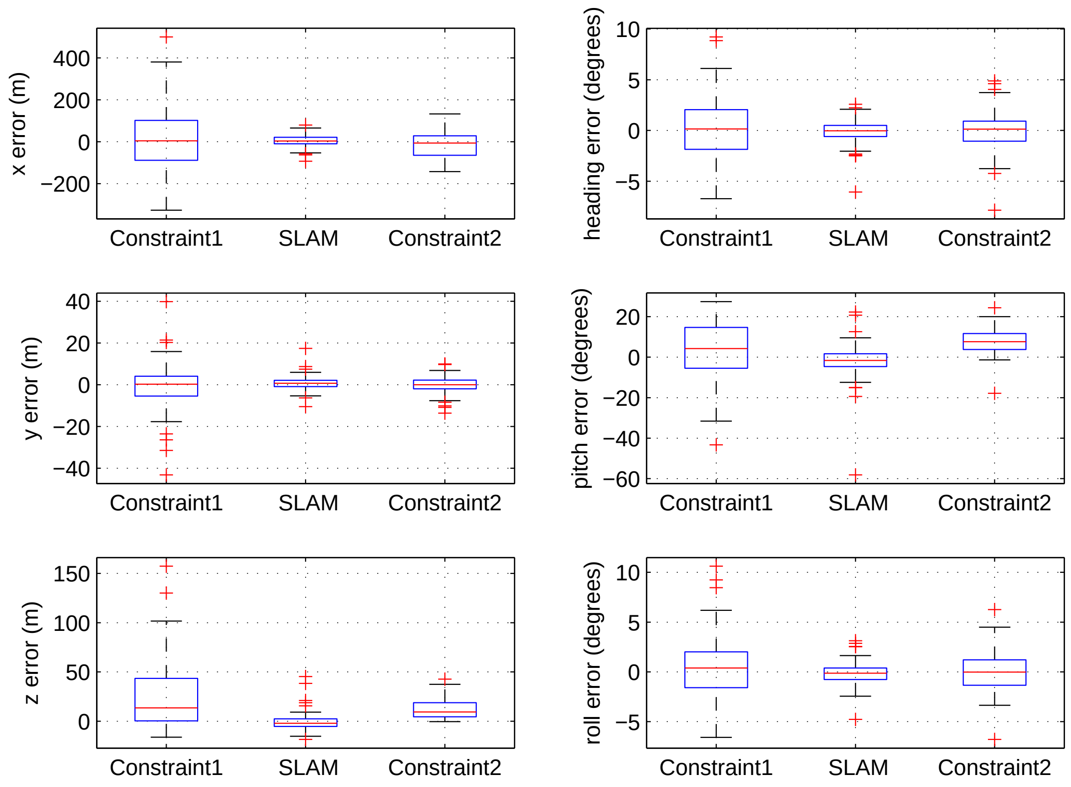


Fig. 17. Comparative results between fusion techniques – the straight line, fixed camera flight

across all i and j, we can rewrite Equation (75) as

$$\begin{aligned} MSE(\tilde{\theta}) &= T_s^2 E\left[\sum_{i=1}^{k} \mathcal{N}^2(0, \sigma_{w,g}^2)\right] + T_s^2 E\left[\sum_{i=1}^{k}\sum_{j=1}^{k} b_g^2\right] \\ &= T_s^2 k \sigma_{w,g}^2 + T_s^2 k^2 b_g^2/2 \qquad (76) \\ &= \mathcal{D}(t^2\sigma_{b,g}^2) + \mathcal{D}(t\sigma_{w,g}^2). \end{aligned}$$

This equation shows that the white noise causes the mean squared error of the attitude estimate to increase at a rate proportional to time ($t = T_s k$), while the bias noise causes a squared increase in error over time. Note that the velocity estimate ($\hat{V}$) will have the same form of error variance, with the subscripts g replaced with a.

2) Complete System Error: The navigation state of our system is represented by an x and y location, the current angle (θ), and the current velocity (V). By integrating a single sensor, the drift order of θ and V was evaluated in the previous sub-section. The x and y location error, however, requires another integration of collected data. Using similar notation as before, the total error in position is

$$\tilde{\vec{p}} = \sqrt{(\hat{x} - x)^2 + (\hat{y} - y)^2}. \qquad (77)$$

To find the mean squared error of this quantity, we first rewrite $\hat{x}$ and $\hat{y}$ as a function of $\hat{\theta}$ and $\hat{V}$, as follows:

$$\hat{x} = \sum_{i=1}^{k} (V_i + \tilde{V}_i) \cos(\theta_i + \tilde{\theta}_i) \qquad (78)$$

$$\hat{y} = \sum_{i=1}^{k} (V_i + \tilde{V}_i) \sin(\theta_i + \tilde{\theta}_i) \qquad (79)$$

$$\tilde{\vec{p}} = \sqrt{(\hat{x} - x)^2 + (\hat{y} - y)^2}. \qquad (80)$$

Squaring the $\tilde{\vec{p}}$ term and applying some trigonometry properties leads to

$$\begin{aligned} \tilde{\vec{p}}^2 = \sum_{i=1}^{k}\sum_{j=1}^{k} \; & V_i V_j \cos(\theta_i - \theta_j)(\cos\tilde{\theta}_i - 1)(\cos\tilde{\theta}_j - 1) + \\ & V_i V_j \cos(\theta_i - \theta_j) \sin\tilde{\theta}_i \sin\tilde{\theta}_j + \\ & 2V_i V_j (\cos\tilde{\theta}_i - 1) \sin\tilde{\theta}_j \sin(\theta_i - \theta_j) + \\ & 2V_i \tilde{V}_j (\cos\tilde{\theta}_i - 1) \cos(\theta_i - \theta_j - \tilde{\theta}_j) + \quad (81) \\ & 2V_i \tilde{V}_j \sin\tilde{\theta}_i \sin(\theta_i - \theta_j - \tilde{\theta}_j) + \\ & \tilde{V}_i \tilde{V}_j \cos(\theta_i - \theta_j + \tilde{\theta}_i - \tilde{\theta}_j). \end{aligned}$$

To simplify this expression, we assume $\theta_i = \theta_j$, leading to

$$\begin{aligned}\tilde{\vec{p}}^2 &= \sum_{i=1}^{k}\sum_{j=1}^{k} V_iV_j\cos(\tilde{\theta}_i-\tilde{\theta}_j)+V_iV_j+\\ &\quad -V_iV_j\cos\tilde{\theta}_i - V_iV_j\cos\tilde{\theta}_j+\\ &\quad 2V_i\tilde{V}_j\cos(\tilde{\theta}_i-\tilde{\theta}_j)+\\ &\quad -2V_i\tilde{V}_j\cos\tilde{\theta}_j+\\ &\quad \tilde{V}_i\tilde{V}_j\cos(\tilde{\theta}_i-\tilde{\theta}_j).\end{aligned}$$

Taking the expected value and using the fact that all noises are uncorrelated with each other leads to:

$$\begin{aligned}\tilde{\vec{p}}^2 &= \sum_{i=1}^{k}\sum_{j=1}^{k} E[V_iV_j]E[\cos(\tilde{\theta}_i-\tilde{\theta}_j)]+E[V_iV_j]+\\ &\quad -E[V_iV_j]E[\cos\tilde{\theta}_i]-E[V_iV_j]E[\cos\tilde{\theta}_j]+\\ &\quad 2E[V_i\tilde{V}_j]E[\cos(\tilde{\theta}_i-\tilde{\theta}_j)]+\\ &\quad -2E[V_i\tilde{V}_j]E[\cos\tilde{\theta}_j]+\\ &\quad E[\tilde{V}_i\tilde{V}_j]E[\cos(\tilde{\theta}_i-\tilde{\theta}_j)].\end{aligned} \tag{82}$$

This equation can be divided into 4 different terms, each of which can be handled separately as follows:

- $E[V_iV_j]$ is the expected value of the true velocity values. For mathematical simplicity, we assume that the velocity is constant, so $E[V_iV_j]=V^2$.
- $E[V_i\tilde{V}_j]$ will become zero as there is assumed to be no correlation between the true velocity values (V_i) and the and the noise in velocity estimates ($\tilde{V}_j$).
- $E[\tilde{V}_i\tilde{V}_j]$ is the cross-correlation between the errors at different times i and j. Because these values are obtained by integrating the accelerometer outputs, there are two types of correlation. First, the correlated noise due to integrated white noise will be of the form $\mathcal{N}(0,\min(i,j)\sigma^2_{w,a})$. The noise due to bias in the accelerometer will be ijb_a.
- $E\left[\cos\left(\mathcal{N}(0,\sigma^2)\right)\right]$ appears repeatedly in Equation (82), with the normal distribution changing depending on the error term inside the cos. As discussed in Appendix A, we can replace this expected value with the term $e^{-\sigma^2/2}$. Note that the value of σ^2 will depend on which expected value term in Equation (82) is being replaced.

Following these rules for replacing the expected values leads to the equation

$$\begin{aligned}E[\tilde{\vec{p}}^2] &= \sum_{i=1}^{k}\sum_{j=1}^{k} V_iV_je^{-(T_s^2(i-j)^2b_g^2+T_s^2(i-j)\sigma^2_{w,g})/2}+V_iV_j+\\ &\quad -V_iV_je^{-(T_s^2i\sigma^2_{w,g}+T_s^2i^2b_g)/2}+\\ &\quad -V_iV_je^{-(T_s^2j\sigma^2_{w,g}+T_s^2j^2b_g)/2}+\\ &\quad (\min(i,j)\sigma^2_{w,a}+ijb_a)e^{-\sigma^2_{cross}/2}\\ \sigma^2_{cross} &= T_s^2(i-j)^2b_g^2+T_s^2(i-j)\sigma^2_{w,g}\end{aligned}$$

Before we can formally derive the drift order for this expression, more attention must be given to the exponential terms. Note that each exponential term could be rewritten in the form $k_1e^{-k_2(x-\mu)^2/2}$, where k_1 and k_2 are constants that are greater than 0. Noting the similarity between this expression and a typical Gaussian distribution, we assert that the infinite sum of any of these exponential terms will converge to a constant. Therefore, we can divide the equation above into navigation order components as follows (replacing each exponential with a constant):

$$\begin{aligned}\sum_{i=1}^{k}\sum_{j=1}^{k}V_iV_j &= \mathcal{D}(t^2)\\ \sum_{i=1}^{k}\sum_{j=1}^{k}V_iV_jk_1e^{-k_2(x-\mu)^2/2} &= \mathcal{D}(t)\\ \sum_{i=1}^{k}\sum_{j=1}^{k}\min(i,j)\sigma^2_{w,a}k_1e^{-k_2(x-\mu)^2/2} &= \mathcal{D}(t^2)\\ \sum_{i=1}^{k}\sum_{j=1}^{k}ijb_ak_1e^{-k_2(x-\mu)^2/2} &= \mathcal{D}(t^3)\end{aligned} \tag{83}$$

Therefore, the MSE of location increases in proportion to t^3. Of particular note in this derivation is the role that noise in the angular estimate plays in the drift order of the system. In the last line of Equation (83), the effects of angular error are simply a constant which is ignored by the drift order analysis. Therefore, *white and bias errors in the gyroscope do not affect the location drift order of this system.*

B. Performance of epipolar-constraint based fusion

One of the methods previously introduced in the literature to enable fusion between visual and inertial information is based on the "epipolar" constraint[87], [63]. In this section, we review how the epipolar constraint is used to enable fusion of visual and inertial information, followed by a discussion of how this method affects the drift order of the system.

1) Utilizing the epipolar constraint: The epipolar constraint can be utilized whenever a single fixed object is observed by a camera at two locations (or two cameras at different locations). Given that any three points in the world form a plane, a single world point and the two camera projection centers form a plane in the 3-D world – the *epipolar* plane. Similarly, when a world point is observed in two images, the two vectors representing where the point was observed in the image plane ($\vec{x'}$ and $\vec{x}$), and the translation vector between the two camera locations, will all lie on the epipolar plane.[6] By assigning a unique coordinate frame to each camera location, this constraint is represented by the formula

$$\vec{x'}[\vec{p}^{c2}_{c1}]_\times\mathbf{C}^{c2}_{c1}\vec{x}=0, \tag{84}$$

[6]Note that image locations are typically in pixels, while the discussion of vectors so far assumes all vectors are in an unscaled, Euclidean space. In this report, we assume that the camera has been calibrated a-priori and that the conversion from image to Euclidean vectors has already occurred using this calibration information.

where $\mathbf{C}_{c1}^{c2}$ is the direction cosine matrix between the camera coordinate frames and $[\vec{p}_{c1}^{c2}]_\times$ is the position of camera 1 in the second camera's coordinate frame, put into a skew-symmetric matrix. (In other words, we calculate the cross product between $\vec{p}_{c1}^{c2}$ and $\mathbf{C}_{c1}^{c2}\vec{x}$.) This constraint enforces that the three vectors $\vec{x'}$, $\mathbf{C}_{c1}^{c2}\vec{x}$, and $\vec{p}_{c1}^{c2}$ all lie within the same plane in the 3-D world.

To utilize the epipolar constraint in a fusion environment, we created a Kalman Filter-based framework. (Specifically, we utilized the Unscented Kalman Filter (UKF) to apply the Kalman filter to non-linear systems.) The state in the UKF must contain enough information to generate both $\mathbf{C}_{c1}^{c2}$ and $\vec{p}_{c1}^{c2}$ during the measurement step. To represent general motion by the robot between two different locations, the UKF state $\vec{X} = [\chi_t, \chi_{t-1}, b]^T$ is used, where χ_t is the navigation state estimate at time t, χ_{t-1} is the navigation state estimate at the previous time, and b is the estimated bias in the accelerometer and gyroscope. The navigation state (χ) at each time contains x, y, θ, V as described in the prior section on IMU-only navigation. This method for setting up the UKF to enable vision and inertial information fusion was first introduced in [65].

Performing the time update

The time update for this UKF implementation takes two different forms. The first form updates χ_t (the first 4 elements of the current state) every time an IMU measurement occurs using the same dynamic equations as the IMU-only estimator. This makes the first 4 elements of the state the most recent navigation state estimate. After each *measurement* update, the state is also updated using the formula

$$\vec{X}^+ = \mathbf{A}\vec{X}^- \text{ where} \tag{85}$$

$$\mathbf{A} = \begin{bmatrix} I & 0 & 0 \\ I & 0 & 0 \\ 0 & 0 & I \end{bmatrix}, \tag{86}$$

causing the current state to become $\vec{X} = \begin{bmatrix} \chi_t \\ \chi_t \\ b \end{bmatrix}$. The first 4 elements of the state vector are then updated by IMU measurements to realize a new current state $\mathbf{X} = \begin{bmatrix} \chi_{t+1} \\ \chi_t \\ b \end{bmatrix}$. With this technique, the two most recent χ estimates, corresponding to the time of the two most recently captured images, are always stored as the current state, together with the estimated biases for the gyroscopes and accelerometers.

Performing the measurement update

To utilize the epipolar constraint as a measurement in a UKF framework, the "Dynamic Vision" approach introduced in [63] is used. Assuming a feature has been detected in two images, the locations of the features are represented by $\vec{x'}$ and $\vec{x}$. Because the epipolar constraint should always be equal to zero, the "measurement" used by the UKF is a vector of zeros in length equal to the number of corresponding features found between the two images. The predicted measurement is $\vec{x'}[\vec{p}_{c1}^{c2}]_\times \mathbf{C}_{c1}^{c2}\vec{x}$ (from Equation (84)) for each set of features $\vec{x'}$ and $\vec{x}$, where $\vec{p}_{c1}^{c2}$ and $\mathbf{C}_{c1}^{c2}$ are functions of χ_t and χ_{t-1}.

2) Effect of epipolar-based fusion: We perform our analysis of the drift order in two portions. First, we analyze the effect of epipolar-based fusion on the error in $\hat{\theta}$. Second, we analyze the effect on the drift order of the location error.

Drift order of attitude error

To understand the impact of epipolar-based fusion on the drift order of angular estimates, we analyze the measurement term $\vec{x'}[\vec{p}_{c1}^{c2}]_\times \mathbf{C}_{c1}^{c2}\vec{x}$. Note that the measurement includes information about the attitude change between the two cameras ($c1$ and $c2$), but does not contain any information about the actual attitude of the camera. Because the epipolar measurements do not return a measurement of absolute attitude, and the measurement of relative attitude is noisy, it cannot wholly eliminate the drift in $\hat{\theta}$. However, it will reduce the drift order of the θ error, as shown below.

Theorem 1: By obtaining relative measurements of attitude, the bias of the gyroscope can be successfully estimated, reducing the drift order of $\hat{\theta}$ to $\mathcal{D}(t)$ from $\mathcal{D}(t^2)$.

Proof To prove that the bias of the gyroscope can be properly estimated when using relative measurements, we will use observability theory on a simplified system. The system state is $[x_n, x_o, b]^T$, where x_n is the new state, x_o is the old state, and b is the estimate bias in a sensor that is used to update x_n. The time update for this equation is

$$\begin{aligned} \vec{x}_{t+1} &= \mathbf{F}\vec{x}_t + \mathbf{G}u \text{ where} \\ \mathbf{F} &= \begin{bmatrix} 1 & 0 & -\Delta t \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ \mathbf{G} &= [\Delta t \quad 0 \quad 0]^T, \end{aligned}$$

where u is a scalar input from a rate sensor. The measurement equation is $y = \mathbf{H}\vec{x}$ where $\mathbf{H} = [1, -1, 0]^T$. The observability matrix for this system is

$$\begin{aligned} \mathcal{O} &= \begin{bmatrix} \mathbf{H} \\ \mathbf{HF} \\ \mathbf{HFF} \end{bmatrix} \\ &= \begin{bmatrix} 1 & -1 & 0 \\ 1 & -1 & -\Delta t \\ 1 & -1 & -2\Delta t \end{bmatrix}. \end{aligned}$$

From inspection, the row-space spanning vectors for this matrix include $[1, -1, 0]$ and $[0, 0, 1]$. The second vector proves that the biases are observable for a system in which only relative differences over time are observed, showing that the gyroscope bias will be estimated when using epipolar-based fusion of inertial and visual data.

Because the biases are observable, this should decrease the drift order on $\hat{\theta}$ from $\mathcal{D}(t^2)$ to $\mathcal{D}(t)$.

Drift order of location error

To analyze the drift order on location error, we once again consider the epipolar constraint $\vec{x'}[\vec{p}_{c1}^{c2}]_\times \mathbf{C}_{c1}^{c2}\vec{x} = 0$. Note that because this equation is set equal to 0, no information about the magnitude of relative position $\vec{p}_{c1}^{c2}$ can be obtained from

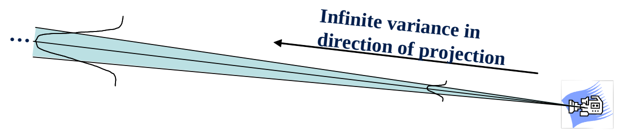


Fig. 18. A simple figure demonstrating the extremely non-linear variance results from a single camera observation of a point. First, there is infinite variance in the direction of the observation. Second, as the distance from the camera increases, the variance in orthogonal directions also increase

epipolar measurements. Therefore, the epipolar constraint will have no effect on the velocity estimate error. Because of the property of drift order described at the end of Section VI-A2, namely, that the location drift order is independent of noise in the gyroscopes, *the drift order on location is the same when using an epipolar-based fusion and an IMU-only navigation method.*

C. Performance of SLAM-based fusion methods

To explore the performance of SLAM-based methods, we first review the basics of SLAM-based algorithms. The general concept of SLAM is that a set of world points should be observed (mapped) and localized over time while simultaneously determining the current location of the robot (localization). By maintaining a full covariance matrix between the location of the sensor and the location of the world points, the SLAM algorithm has been shown to be convergent in specific circumstances [77]. Therefore, SLAM has been an area of intensive research for the past several years.

The implementation of SLAM which we discuss in this section is once again based on the Kalman Filter (UKF). The state of our UKF includes the current navigation state of the camera (x, y, θ, V), the biases of the IMU (b), followed by entries for the world points that are being tracked. One of the primary difficulties with visual SLAM is the initialization of new features. Because only the bearing of a world object from the camera is observed the first time, the 3-D location of the world point cannot be added directly to UKF state for two reasons. First, there is essentially infinite variance in one direction if only one observation has occurred (see Figure 18). Second, because there may be error in the observed bearing, the variance in the directions orthogonal to the infinite variance direction grow with distance from the camera, causing the initial observation of the point to have an extremely non-linear variance.

To provide an initialization methodology that can be used with a single camera and does not pre-suppose information about the target, we use the method introduced in [85] to initialize feature locations. This method, the "inverse depth" method, places three new items in the UKF state for each new object that is observed, namely: (1) the projective center of the camera when the feature was first observed (represented by P_c), (2) the direction of the ray (in 3-space) of the first observation (represented by two angles, θ and ϕ), and (3) the inverse depth of the point along that line (represented by ξ). The inverse depth, rather than the depth, is used because a finite value between 0 and 1 covers the entire depth from 1 to infinity in a linear fashion. Therefore, a variance of .5 on the inverse depth is equivalent to an infinite variance when using depth. In addition, because the bearing of the original observation is included in the Kalman Filter state, the "increasing variance with distance" problem discussed above is properly represented in the Kalman Filter.

With the state vector described, we can now describe in detail the time and measurement updates utilized in our SLAM-based fusion technique. During the time update, only the current navigation state estimate of the camera needs to be updated. The updates occur according to the accelerometer and gyro measurements from the IMU. During the measurement step of the UKF, the location of each world point that is currently being observed in the image is used. The predicted measurement for each world location is

$$\vec{x} = \lambda \mathbf{C}_n^c (\vec{P}_c^n + \frac{1}{\xi}\vec{r} - \vec{p}_c^n), \tag{87}$$

where $\mathbf{C}_n^c, \vec{P}_c^n, \xi, \vec{r}$, and $\vec{p}_c^n$ are all derived from the current state estimate. λ is a simple scaling factor which divides the results of $\mathbf{C}_n^c(\vec{P}_c^n + \frac{1}{\xi}\vec{r} - \vec{p}_c^n)$ to make the third element of $\vec{x} = 1$.

1) Drift order of SLAM-based systems: In prior work [77], [78], several different evaluations of the performance of SLAM-based algorithms have been performed. The primary result of these works is that SLAM is *convergent* (i.e., the drift rate goes to zero as $t \rightarrow \infty$.) However, the analysis showing convergence assumes that the robot observing features is always observing the same set of features, limiting the use of the robot to a small area. In this section, we present an analysis to determine the effect of the inclusion and removal of features at a constant rate (the *feature replacement rate* – frr). This new analysis enables us to characterize navigation performance when the robot is navigating over an extremely large area.

To analyze SLAM algorithms with feature replacement, we analyze the performance of SLAM in three separate "stages". First, we analyze SLAM when the old feature is still being observed, but the new feature has not yet been observed. Second, we analyze the case when a new feature is being observed simultaneously with the old feature. Third, the performance of SLAM with the new feature after the old feature is no longer being observed is analyzed. To help guide the explanation, we will use the simple setup shown in Figure 19 where the robot moves in a single line, the features are also in a line, and the sensor measures the distance between a feature and the robot. This leads to our first lemma, which is:

Lemma 1: When a SLAM algorithm has converged, the feature location is known with uncertainty σ_f^2.

This lemma is a direct result of prior work on convergence. The SLAM algorithm has converged when the feature location errors are no longer increasing and are highly correlated

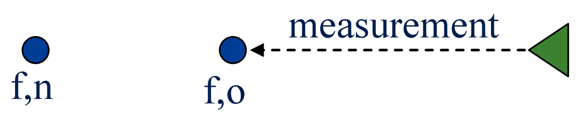


Fig. 19. This figure shows a simple scenario that demonstrates the essential characteristics in SLAM with feature replacement. The robot (the triangle) observes the "old" feature first (f,o), later observes both features, and finally observes only the "new" feature (f,n).

with the current robot position. This lemma leads to our first theorem relating to SLAM-based fusion, which states:

Theorem 2: The uncertainty in the robot location when consistently observing a single feature is bounded by $\sigma_f^2 + k\sigma_m^2$, where σ_f^2 is the variance of the feature, σ_m^2 is the variance associated with the measurement itself, and k is a constant such that $0 \le k \le 1$.

Proof: From Lemma 1, we know that if the robot location were completely unknown at a given point in time, its covariance would decrease to $\sigma_f^2 + \sigma_m^2$ immediately following a measurement of the feature. This forms the upper limit on the covariance of robot location, proving $k \le 1$. In addition, from lemma 1, the error in feature location is correlated with the robot location, leading to the lower limit of σ_f^2. k will usually be somewhere between 0 and 1 because the robot location is not entirely unknown before each measurement occurs. Instead, it is a function of the previous uncertainty and how quickly the location estimate drifts (σ_d^2). Therefore, at each measurement, the covariance due to measurement will decrease, finally reaching an equilibrium point that is dependent on the relative sizes of σ_d^2 and σ_m^2. Therefore, the final uncertainty in robot location will be $\sigma_f^2 + k\sigma_m^2$, where k depends on the relative magnitudes of σ_d^2 and σ_m^2 and is bounded by $0 \le k \le 1$.

Let us now transition to the second stage in feature replacement: the case where a new feature is added to the SLAM filter. We will assume that the old feature is not immediately removed, but that the new feature and old feature are simultaneously observed for some time period we term the *joint observation time* (jot). To determine the final drift order of the system, we must characterize what the uncertainty of the new feature location will be. To determine this new covariance, we introduce the notation $\sigma_{f,o}^2$ and $\sigma_{f,n}^2$ for the covariance of the old and new features, respectively.

Theorem 3: The covariance on a new feature will be bounded by $\sigma_{f,o}^2 + \frac{(1+k)\sigma_m^2}{jot}$.

Proof: From theorem 2, we know that covariance of the robot location is $\sigma_{f,o}^2 + k\sigma_m^2$. Therefore, the covariance for the new landmark after the first measurement is $\sigma_{f,o} + (1+k)\sigma_m^2$. Because the $\sigma_{f,o}$ term is present in every measurement and correlated with both robot location and the old feature location, it will not decrease with extra measurements of the new feature. The $(1+k)\sigma_m^2$ term, however, is not correlated with robot location and will therefore be reduced by every measurement of the new feature. Therefore, the covariance of the new feature will be the summation of $\sigma_{f,o}^2$ and $(1+k)\sigma_m^2$ divided by the number of observations of the new feature. Assuming the number of observations of the new feature are proportional to the time spent observing it (jot), we obtain $\sigma_{f,n}^2 = \sigma_{f,o}^2 + \frac{(1+k)\sigma_m^2}{jot}$. QED.

Finally, we must analyze what happens after the old feature is no longer observed by the SLAM filter. This case is very similar to that discussed in theorem 2 where a single feature is being used to maintain the position of the robot. Substituting in the noise on the new feature location with the expression described in theorem 3, we can express the new robot location variance as a function of the first feature's variance. This leads to $\sigma_l^2 = k\sigma_m^2 + \sigma_{f,o}^2 + \frac{(1+k)\sigma_m^2}{jot}$. Note that with each feature replacement, the final term will be added again to the MSE of the robot location. Therefore, when running SLAM with feature replacement, the navigation order is of the form $\mathcal{D}(t)$, where the constant inside this expression will be multiplied by $\frac{frr}{jot}$.

D. Results

In the preceding sections, we have analytically derived the location drift order (in terms of mean square error) for IMU-only navigation ($\mathcal{D}(t^3)$), epipolar-based fusion (also $\mathcal{D}(t^3)$) and SLAM-based fusion ($\mathcal{D}(t)$). To verify our results, we created a Simulink-based environment consisting of (1) a robot with an IMU, (2) world features, (3) a camera model, and (4) three estimators of navigation state (IMU-only, epipolar-based fusion, and SLAM-based fusion). The robot was defined by the state equations

$$\dot{x} = V\cos\theta \tag{88}$$
$$\dot{y} = V\sin\theta \tag{89}$$
$$\dot{\theta} = \omega \tag{90}$$
$$\dot{V} = a, \tag{91}$$

with ω and a being inputs to the simulation environment. The IMU was modeled by corrupting ω and a with white noise and a bias. At the beginning of each simulation run, a set of 51 features was uniformly distributed throughout the simulated environment of 3000x3000 units. To determine the altitude of each feature, a Gaussian distribution with standard deviation of 200 was used. The camera was limited to observe features within $\pm 60^o$ side to side and $\pm 50^o$ top to bottom. The feature location (f), when observed, was defined by the equation

$$\begin{aligned}\begin{bmatrix} x \\ y \\ z \end{bmatrix} &= C(X-T) \\ f &= \begin{bmatrix} x/z \\ y/z \end{bmatrix}. \end{aligned} \tag{92}$$

The feature locations were then corrupted with white, Gaussian noise. This simulates the use of a calibrated camera without radial distortion in the real-world. Finally, the three estimators described in the prior sections were implemented,

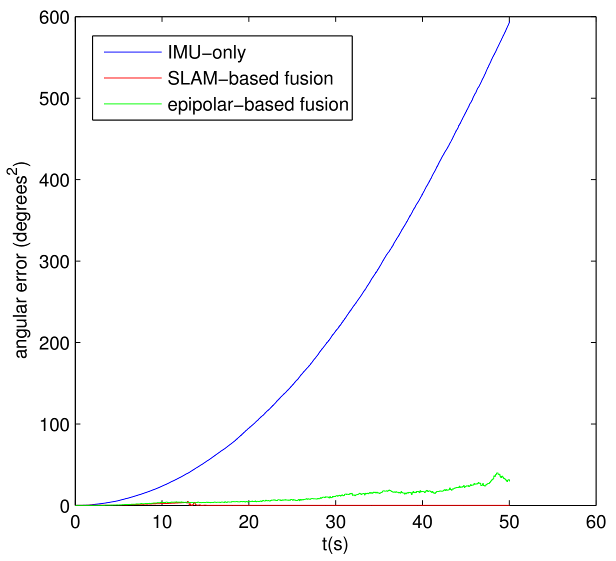


Fig. 20. This plot shows the MSE of the angle estimates of three different estimators. Both the epipolar and SLAM-based fusion techniques significantly reduce the angular error.

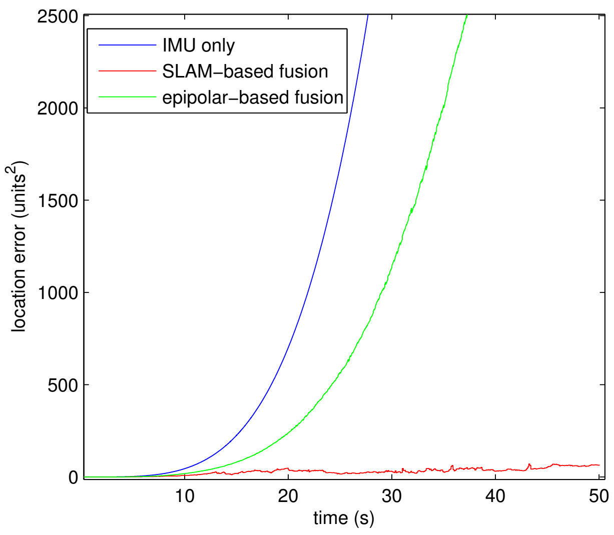


Fig. 21. This plot shows the MSE of the location estimates of the three different estimators. While the epipolar-based constraint significantly lowers the angular error (see Figure 20), its drift order is the same as IMU-only estimates. SLAM-based techniques increase very slowly (linearly with time) as predicted by our analysis.

each having the same IMU and feature inputs so that a direct comparison between techniques is achieved.

In Figure 20, we present the MSE of each estimator on $\hat{\theta}$. The plots represent the MSE across 10 runs of the simulator. Each simulation consisted of the robot moving at a constant velocity of 13 units/s and an angular rate of 0.05 rad/s. The white and bias noise on the accelerometer was added as a Gaussian distribution with standard deviation of .1 units/s^2. The gyroscope error was models with Gaussian distributions with a standard deviation of 0.01 rad/s. Note in Figure 20 how the IMU-only curve shows drift in MSE at the rate of t^2, verifying our analysis of the IMU-only case in section VI-A1. Both the epipolar and SLAM-based fusion techniques dramatically decrease the error in the attitude estimate.

In Figure 21, the MSE of location error for the three possible techniques are also shown. These plots are averaged over the same 10 runs described for the errors in θ. Both the IMU-only and epipolar-based fusion approaches grow in error at the rate of t^3, verifying the analytical results presented above that state the epipolar-based and IMU-only methods are of the same drift order. SLAM-based fusion, on the other hand, is increasing at a (small) linear rate.

Figures 20 and 21 also verify the surprising result presented at the end of Section VI-A2: that errors in attitude do not affect the location drift order of the system. Note in 20 the dramatic improvement in errors in theta for the epipolar-based fusion approach. However, note in Figure 21 that the *drift order* of location for the epipolar-based approach is no different from the IMU-only approach. This empirically verifies that the drift rate of attitude does not affect the drift rate of location error.

VII. Reduced-drift navigation

Nomenclature

x^i, y^i	Position of the i_{th} MAV
V^i	Velocity of the i_{th} MAV
Ψ^i	Heading of the i_{th} MAV
Φ^i	Roll angle of the i_{th} MAV
v_x^i	Velocity in X direction m/s
v_y^i	Velocity in Y direction m/s
a_x^i	Acceleration in X direction, m/s^2
a_y^i	Acceleration in Y direction, m/s^2
g	Acceleration due to gravity, m/s^2
R_{ij}	Relative range between i_{th} and j_{th} MAV
ϕ_{ij}	Relative bearing angle between i_{th} and j_{th} MAV
β	Side slip angle
b_Φ^i	gyro bias for roll rate
b_Ψ^i	gyro bias for yaw rate

In the prior sections we have discussed methods for improving the navigation of a single MAV. For the remained of this report, we discuss techniques developed for multiple MAV missions. Cooperative missions have been studied extensively for reasons unrelated to navigation. For example, some MAV missions may be quite dangerous, and it is unlikely that a single agent would survive long enough to complete the task. Similarly, some MAV models are inexpensive but prone to failure, in which case it may be more economically feasible to use a large number of cheap MAVs rather than risk a single expensive one. Many tasks, such as searching a particular area, can be completed more quickly using multiple MAVs[92].

1) Related Work: Cooperative techniques in the past have generally focused on creating basic group behaviors, including: (1) remaining in the group (cohesion), (2) keeping a safe distance between each other (dispersion), (3) alignment in the direction of motion, (4) collision avoidance from obstacles and moving MAVs [93], and (5) cooperative arrival[94], [95]. In most of these works, the MAVs all navigate individually, followed by a step to ensure group behavior. In this section, however, we show that MAVs can cooperatively navigate by helping each other to determine position and heading when GPS is not available. We have found that a cooperative navigation strategy significantly constrains the IMU drift, similar to the results of a hybrid IMU/GPS system.

In Kurazume and Hirose[96], [97], a method called "Cooperative Positioning System (CPS) is described. In this system, a group of robots is divided into two groups, A and B. One group, A, remains stationary and acts as a landmark while group B moves. Group B then stops and acts as a landmark for group A. This "dance" is repeated until the target position is reached. These kind of strategies, however, cannot be applied to fixed-wing MAVs because they require continuous motion to stay in the flight. In another paper [98], the authors study the trade offs between different classes of sensing strategy and motion control strategy in the context of terrain mapping with multiple robots. However, this work also assumes the ability of robots to stop, making it inapplicable to fixed-wing MAVs.

In Merino et al.[99], a method for vision based MAV motion estimation from multiple planar homographies has been proposed. The authors describe the determination of the relative displacement between different MAVs employing techniques for blob feature extraction and matching. This work, however, requires that the MAVs be capable of identying and tracking objects in the surrounding world, while our work depends solely on the group of MAVs in use for the mission.

In Roumeliotis and Bekey[100], a "collaborative localization" scheme is proposed that enables robots with different sensors and communication ranges to improve their pose estimate when they are within communication range. While similar, our work is different from Roumeliotis and Bekey in two principle ways. First, Roumeliotis and Bekey assume that the robots can use external landmarks to help navigate the robots. We limit the navigation system inputs to IMU measurements and relative information between the MAVs. Second, we perform and observability analysis of our cooperative navigation system, yielding insight into the benefits and limitations of cooperative navigation.

2) Contribution: In this section, a cooperative navigation system (CNS) is outlined that allows an MAV within a group to navigate without GPS or other wide-area positioning information. This method works on the principle of exchanging IMU information within the group of MAVs. An MAV collects the IMU measurements from other MAVs in its sensor range and then fuses these measurements with the range and bearing measurement of neighboring MAVs. This fusion results in an estimate of the position and attitude state for itself and other MAVs in its sensor range. Simulation results demonstrate the ability to significantly limit drift in the navigation states using this method. We also perform an observability analysis to help define the strengths and limitations of CNS.

In the next section we describe the full cooperative navigation model and perform an observability analysis. In section VII-B, simulation results are presented that demonstrate the efficacy of the CNS.

A. Cooperative Navigation System

To develop the CNS we use a simplified model where the MAV is moving in level flight and the velocity of the MAV is aligned with the wind frame of the MAV ($\beta = 0$). We also assume that each MAV in the group is equipped with a low cost IMU for its navigation, which consist of two gyroscopes (gyros) and two accelerometers. The gyros gives roll rate($\dot{\Phi}$) and yaw rate. The two accelerometers give acceleration a_x and a_y as output. Due to the low-cost and low-weight restrictions of MAVs, the IMUs present on the MAV are not sufficient to enable navigation by simply integrating the gyro and accelerometer inputs. Therefore, GPS is typically used. We assume, however, that none of the MAVs have GPS. Instead, a sensor which detects the relative range (R) and bearing (ϕ) of the other MAVs is present to aid in navigation (see Figure 22). In addition, we assume that MAVs have the communication capabilities through which they can exchange the measured accelerometer and gyro data with each other.

Once the IMU information from all MAVs has been exchanged, and the relative location of other MAVs is measured, an Extended Kalman Filter (EKF) is used to fuse together this information. In the following subsections, we describe our dynamic model for the MAVs, followed by a detailed description of the time and measurement update of the EKF.

1) MAV Dynamic Model: To create an EKF for MAV position and attitude, we use a conventional dynamic model for an MAV moving in level flight. The MAV uses two control inputs, the first is commanded roll angle Φ_C, responsible for lateral acceleration and the second is required velocity V_C. Assuming these constraints, the equations of motion for the ith MAV are given as:

$$
\begin{aligned}
\dot{x}^i &= V^i \cos(\Psi^i) && (93)\\
\dot{y}^i &= V^i \sin(\Psi^i) && (94)\\
\dot{\Psi}^i &= \frac{g}{V^i} \tan(\Phi^i) && (95)\\
\dot{\Phi}^i &= \frac{1}{\tau_\Phi^i}(\Phi_C^i - \Phi^i) && (96)\\
\dot{V}^i &= \frac{1}{\tau_V^i}(V_C^i - V^i) && (97)
\end{aligned}
$$

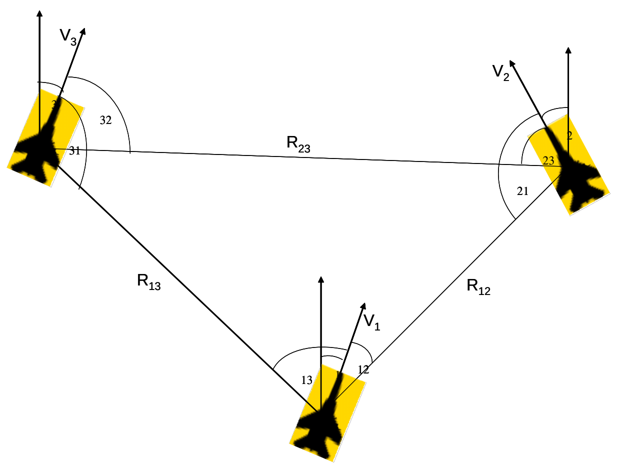


Fig. 22. Estimation Model of MAVs

Where τ_{Φ}^i and τ_V^i are autopilot time constant for roll angle and velocity respectively. Differentiating $\dot{x}^i$ and $\dot{y}^i$ with respect to time we get.

$$\ddot{x}^i = \dot{V}^i\cos(\Psi) - V\sin(\Psi)\dot{\Psi}^i \tag{98}$$
$$\ddot{y}^i = \dot{V}^i\sin(\Psi) + V\cos(\Psi)\dot{\Psi}^i \tag{99}$$

Now substituting $\dot{\Psi}^i = \frac{g}{V^i}\tan(\Phi^i)$, we get.

$$\ddot{x}^i = \dot{V}^i\cos(\Psi) - g\sin(\Psi^i)\tan(\Phi^i) \tag{100}$$
$$\ddot{y}^i = \dot{V}^i\sin(\Psi) + g\cos(\Psi^i)\tan(\Phi^i) \tag{101}$$

Accelerometer outputs with noise in x and y directions are given as

$$a_x^i = \ddot{x}^i - g\sin(\Psi^i)\tan(\Phi^i) + \eta_{ax}^i \tag{102}$$
$$a_y^i = \ddot{y}^i + g\cos(\Psi^i)\tan(\Phi^i) + \eta_{ay}^i \tag{103}$$

Gyro output with bias and noise is given as

$$\dot{\Psi}_{gyro}^i = \dot{\Psi}^i + b_{\Psi}^i + \eta_{\Phi}^i \tag{104}$$
$$\dot{\Phi}_{gyro}^i = \dot{\Phi}^i + b_{\Phi}^i + \eta_{\Psi}^i \tag{105}$$

Range and bearing sensor on each MAV as shown in Figure 22, gives

$$R_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{106}$$
$$\phi_{ij} = \tan^{-1}\frac{y_j - y_i}{x_j - x_i} - \Psi_i \tag{107}$$

2) Navigation State Estimation: For navigation state estimation we use an EKF. To implement the EKF, we assume the following non-linear state space model.

$$\begin{aligned} \dot{X} &= f(X) + w \qquad (108)\\ Q &= E(ww^T)\\ Z &= h(X) + v\\ R &= E(vv^T) \end{aligned}$$

Size of X is $[8n \times 1]$, and it consists of position, velocity, attitude and gyro biases of itself and other $(n-1)$ MAVs in its sensor range. Z consist of $4n$ IMU measurement and $(n-1)$ range and $(n-1)$ bearing measurements. Size of Z is therefore $[2(n-1)+4n \times 1]$. X and Z in the vector form are given by

$$\begin{aligned} X &= [x^1, y^1, v_x^1, v_y^1, \Psi^1, \Phi^1, b_{\Psi}^1, b_{\Phi}^1,\\ \cdots &\quad , x^n, y^n, v_x^n, v_y^n, \Psi^n, \Phi^n, b_{\Psi}^n, b_{\Phi}^n]^T \qquad (109)\\ Z &= [a_x^1, a_y^1, \dot{\Psi}_{gyro}^1, \dot{\Phi}_{gyro}^1, \cdots, a_x^n, a_y^n,\\ \dot{\Psi}_{gyro}^n &\quad , \dot{\Phi}_{gyro}^n, R_{12}, \phi_{12}, \cdots, R_{1n}, \phi_{1n}]^T \qquad (110) \end{aligned}$$

Both system dynamics and measurement are nonlinear, therefore, both of them are linearized for estimation.

$$F = \frac{\partial f(x)}{\partial x}|_{x=\hat{x}} \tag{111}$$
$$H = \frac{\partial h(x)}{\partial x}|_{x=\hat{x}} \tag{112}$$

State and covariance propagation are given as

$$\overline{x}_k = \hat{x}_{k-1} + \dot{\hat{x}}_{k-1}T_s \tag{113}$$
$$P_k^{(-)} = \Gamma_k P_{k-1}\Gamma_k^T + Q_k \tag{114}$$

where Γ_k is transition matrix given by $I + F_kT_s$, Q_k is the process noise covariance matrix, and T_s is sampling time

The gain of EKF during the measurement is given by

$$K_k = P_k^{(-)}H^T(HP_k^{(-)}H^T + R_k)^{-1} \tag{115}$$

State and Covariance update are given as

$$\hat{x}_k = \overline{x}_k + K_k[Z_k - h(\overline{x}_k)] \tag{116}$$
$$P_k = (I - K_kH)P_k^{(-)} \tag{117}$$

In this section we fused the IMU measurements from all the MAVs in the group with the respective range and bearing measurement taken from onboard range and bearing sensors of a MAV. This fused data is then sent as input to EKF explained above, and navigation states of all the MAVs is estimated within the group.

3) Observability Analysis: In this section we perform observability analysis of the CNS to check which states are observable and which are not. We also study the limitations posed by non-observability of states on the accuracy of the CNS. For understanding purpose we do this analysis for a group of two MAVs with constant velocity, which can easily be extended for multiple MAVs. State and measurement vector for two MAVs is given as

$$\begin{aligned} X &= [x^1, y^1, \Psi^1, \Phi^1, b_{\Psi}^1, b_{\Phi}^1, x^2, y^2, \Psi^2, \Phi^2, b_{\Psi}^2, b_{\Phi}^2]^T\\ Z &= [\dot{\Psi}_{gyro}^1, \dot{\Phi}_{gyro}^1, \dot{\Psi}_{gyro}^2, \dot{\Phi}_{gyro}^2, R_{12}, \phi_{12}]^T \end{aligned}$$

EKF is an asymptotic observer given as,

$$\hat{x}_{k+1} = \Gamma_k\hat{x}_k + K_k(Z_k - H\hat{x}_k) \tag{118}$$

EKF chooses gain K_k which makes $(\Gamma_k - K_kH)$ stable, so that state estimation error $\tilde{x}$ is asymptotically stable.

$$\tilde{x}_{k+1} = (\Gamma_k - K_kH)\tilde{x}_k \tag{119}$$

EKF or any asymptotic observer can choose, gain K_k, so that $(\Gamma_k - K_k H)$ have desired eigenvalues (inside unit circle), if and only if (Γ_k, H) is observable.

A system is observable if states can be distinguished in the state space. Observability of nonlinear systems can be found using Lie derivatives. Let

$$l(x) = \begin{bmatrix} L_f^0(h) \\ \cdots \\ L_f^{n-1}(h) \end{bmatrix} \tag{120}$$

Where

$$\begin{aligned} Z &= h = L_f^0(h) \\ \dot{Z} &= \dot{h} = L_f^1(h) \\ &\cdots \\ Z^{n-1} &= L_f^{n-1}(h) \end{aligned}$$

Observability matrix O using Lie derivatives is computed as

$$O = \frac{\partial l(x)}{\partial x} \tag{121}$$

A system is observable if rank of observability matrix is n. To find observable and unobservable states we compute null space of the observability matrix. Null space of observability matrix O for CNS (with no accelerometer) with two MAVS is given as

$$N(O) = \begin{bmatrix} 0 & 0 & 1 \\ 1 & \frac{x^1-x_2}{y^1-y_2} & -\frac{x^1-x^2}{y^1-y^2} \\ 0 & \frac{1}{y^1-y^2} & -\frac{1}{y^1-y^2} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & \frac{1}{y^1-y^2} & -\frac{1}{y^1-y^2} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{122}$$

Thus postion and heading of both MAVs $(x^1, y^1, \Psi^1, x^2, y^2, \Psi^2)$ is not directly observable, whereas roll angle and gyro biases $(\Phi^1, b_\Psi^1, b_\Phi^1, \Phi^2, b_\Psi^2, b_\Phi^2)$ are not in the null space $N(O)$. Hence, roll angle and biases are observable. To find observable modes we compute the row reduced echelon form of the Observability matrix O, given as.

$$O = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & y^1-y^2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & x^2-x^1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Observable modes computed from row reduced observability matrix (O) are given as.

$$\begin{bmatrix} x^1 - x^2 + (y^1 - y^2)\Psi^2 \\ y^1 - y^2 + (x^2 - x^1)\Psi^2 \\ \Psi^1 - \Psi^2 \\ \Phi^1 \\ b_\Psi^1 \\ b_\Phi^1 \\ \Phi^2 \\ b_\Psi^2 \\ b_\Phi^2 \end{bmatrix} \tag{123}$$

First three modes in (123) shows that position and heading of both MAVs are not directly observable. But it can be seen in (123) that if initial position is known then heading of both MAVs becomes observable. Hence CNS gives accurate estimates of heading with bias and roll angle if initial position of MAVs is known.

Observability analysis can be summarized as (1) if we use only IMU without any range and bearing measurement then the IMU bias is unobserved and error in estimates grows with time (2) when range and bearing observations are fused with IMU output then the bias and relative position and relative heading becomes observable. Therefore, if we start at the known position and heading but unknown biases CNS is still able to estimate navigation states with small error even though covariance error in position and heading is growing. This is possible only if we know initial position and heading.

In this section we detailed observability analysis for the CNS which demonstrates that in the CNS biases, relative position and heading are observable. Due to this fact, the CNS constrains the IMU drift significantly. In order to validate our technique we present simulation results in the next section.

B. Simulation Results

In this section we present simulation results to show (1) that the CNS significantly reduces IMU drift, (2) it reduces drift even if biases are zero, (3) gives reasonable estimates with bearing only measurements, and (4) effect of the number of MAVs observed by CNS on the accuracy of CNS.

To simulate CNS and compare it with an IMU only navigation system, we developed a MATLAB environment implementing the dynamic model of MAV described in section II.A. We randomly select the initial position and initial heading of 5 MAVs, with constant bias in all the gyros of each MAV. Measurement and system noise is assumed to be white, with standard deviation $\sigma_R = 4m$, $\sigma_\phi = 5^o$ and $\sigma_{gyro} = 2^o$.

First we show that the CNS achieves significantly improved navigation state estimates compared to an IMU only system. Figures 23, 24, and 25 demonstrate the location, heading, and velocity estimates, respectively, calculated using CNS and an IMU only system. It can be seen that CNS significantly constrains the drift in X and Y position, heading,

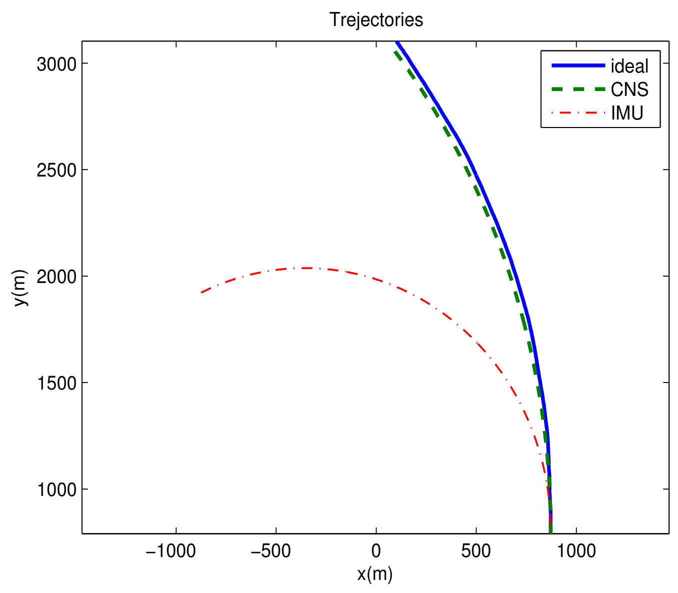


Fig. 23. Trajectory (CNS vs. IMU) for 100s of simulation

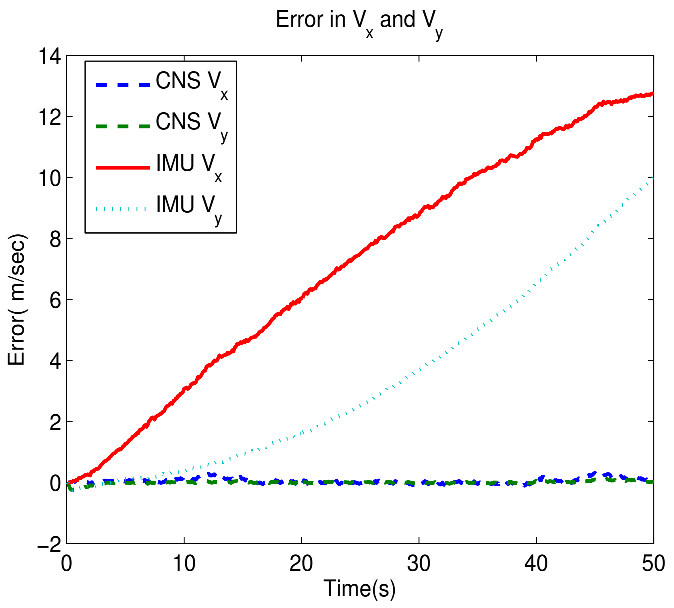


Fig. 25. Error in Velocity(Vx and V_y) CNS vs. IMU(only)

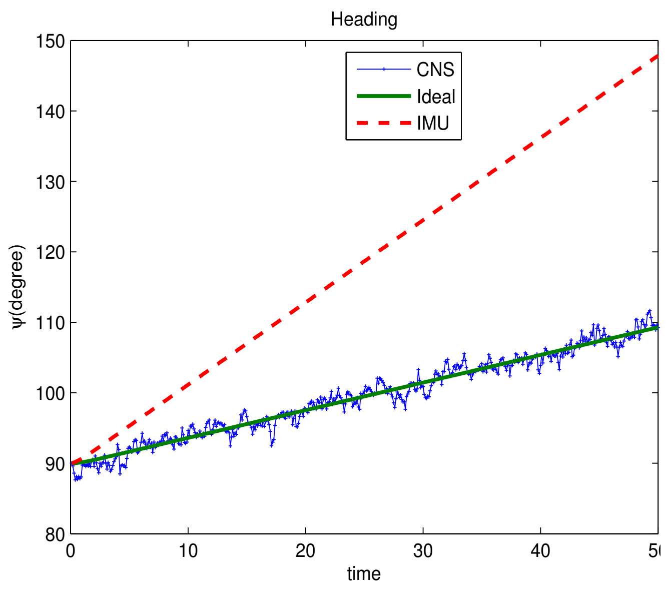


Fig. 24. Heading (CNS vs. IMU) over 100s

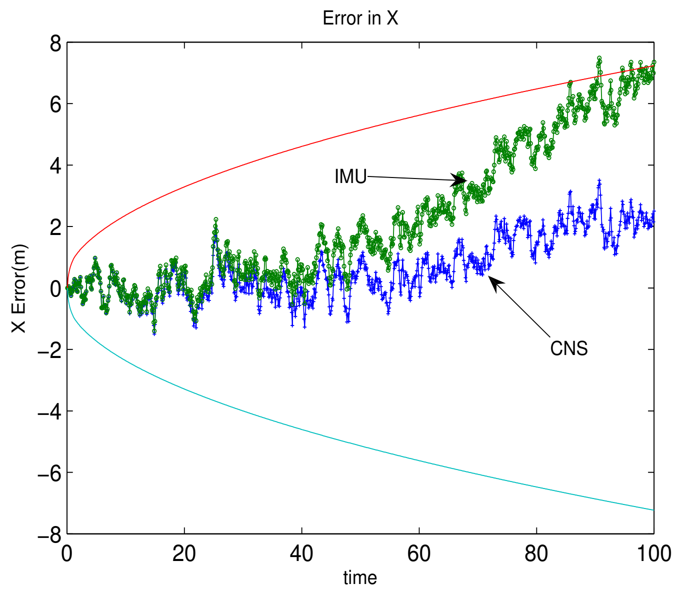


Fig. 26. Error in X CNS vs. IMU(only) with zero bias

and velocity. Note that the majority of the improvement in CNS estimation vs. IMU-only is due to the observability of the IMU biases achieved when using CNS.

Even when the biases are not significant, however, CNS still can decrease the amount of drift compared with an IMU-only system. In Fig 26, we plot the error in X estimates with biases on the gyros set to zero. Note that the error in CNS estimates grwos slower then IMU estimates of the observation of relative location between the MAVs. Hence CNS not only estimates biases accurately but also constrains drift due to noise.

Third, we demonstrate that CNS can be utilized even if range measurements are not available. In practice it is very difficult to have correct range measurements, but bearing measurements can be obtained easily and accurately from a camera. So we simulate the same scenario presented earlier in Fig 23, Fig24, and Fig 25, with no range measurements. In Fig 27 and Fig 28, we present results showing that the error in estimates with bearing-only measurements is higher than when both range and bearing measurements are present. However, the bearing-only scenario still represents at least one order of magnitude improvement over IMU only

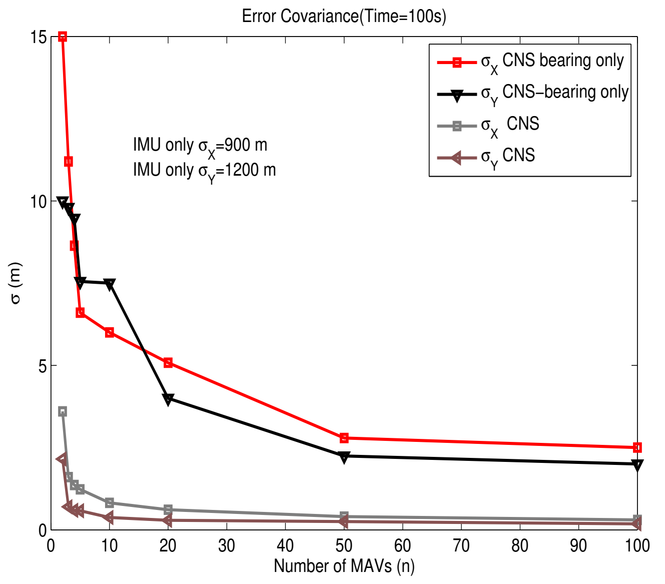


Fig. 27. Uncertainty in X and Y Estimate

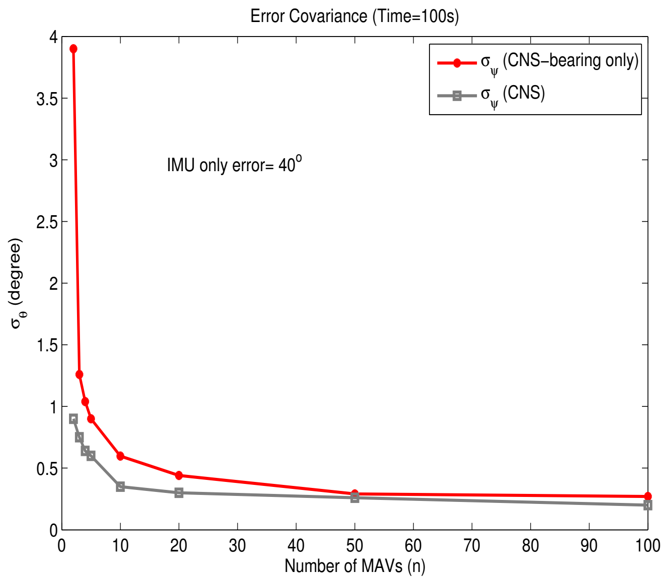


Fig. 28. Uncertainty in heading (Ψ) estimate

navigation.

Finally, simulations are carried out to determine the effect of increasing the number of MAVs that are currently observed by the CNS. Fig 27 shows the uncertainty in X and Y varying with number of MAVs, while Fig 28 shows the uncertainty in heading. Note that uncertainty decreases rapidly with an increase in number of MAVs when the number of MAVs are few, while adding an extra MAV when multiple MAVs are already present has less effect. Because the decrease in uncertainty is more pronounced with few MAVs, the practicality of a CNS with only a few MAVs (5 or less) is demonstrated.

VIII. Drift-free Multi-agent Navigation

In the previous section, we have demonstrated the significant reductions in drift that can be achieved through multi-agent navigation. In this section, we specifically analyze the conditions required for enabling navigation *without drift* in a cooperative setting. We assume that a sparse set of landmarks with known location are available, but that each agent may not be able to observe these landmarks. The exteroceptive sensor used by each agent is a bearing-only sensor such as an electro-optical camera. As a motivating case, consider the scenario of multiple robots exploring a building. If two robots with GPS remain stationary just outside the building, then these robots can be observed as landmarks with known location by other robots. Inside the building, however, there will be walls and other occlusions between the indoor robots and the landmark robots outside the building. The central question addressed in this section is under what conditions can every robot in the system still have drift-free navigation despite occluded views of the landmarks.

The primary tool we utilize to prove conditions for drift-free navigation is non-linear observability analysis. In the past, observability analysis has proven useful when designing navigation systems for single vehicles. Cho et al. [110] use observability analysis to improve the performance of IMU-GPS navigation. IMU and camera calibration is investigated using observability theory in [111]. Observability analysis for simultaneous localization and mapping (SLAM) algorithms is discussed in [112], [113], leading to control methods for maximizing the observability of the system. For multi-agent systems, [114] uses observability analysis to prove the necessary and sufficient number of range measurements to determine relative pose between two agents. Our work is similar to [114] in that we prove necessary and sufficient conditions for determining relative pose between two agents using a bearing-only sensor.

1) Contributions: The primary contributions of this section are:

- Necessary and sufficient conditions for relative pose estimation between two agents using bearing-only sensors.
- The establishment of a direct link between network topology and the rank of the observability matrix.
- Derivation of conditions required to enable drift-free navigation in a multi-agent system.
- A report on real hardware demonstrating the feasibility and practicality of our proposed approach.

2) Section Organization: The rest of the section is organized as follows. Section VIII-A describes our cooperative navigation system. Section VIII-B presents the observability analysis using a graph to represent communication and measurement relationships between agents and derives necessary and sufficient conditions for complete observability. Section VIII-D presents simulation results verifying the analysis.

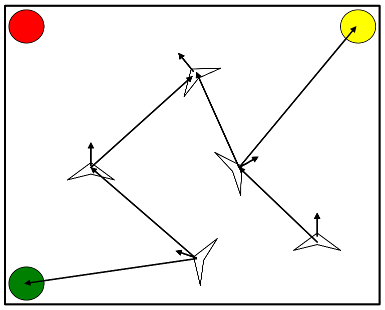


Fig. 29. This figure illustrates the general class of problems addressed in this section. Note that the connections between robots and fixed landmarks are sparse.

Results using real hardware are presented in Section VIII-E.

A. *Bearing Only Cooperative Navigation*

The problem that we consider in this section involves a group of N robots with on-board cameras that can be used to observe the bearing of neighboring robots or of fixed landmarks. The turn rate and velocity of each robot is measured by two wheel encoders. We also assume that the measurements and communications between robots are limited by range. An example of this scenario is shown in Fig. 29, where three fixed landmarks denoted by circles and 5 robots are present in the environment. In addition to showing a distribution of robots and landmarks, Fig. 29 also illustrates the concept of a relative position measurement graph (RPMG). The RPMG is an ordered pair (X, E) consisting of a set of vertices X as robots and the edges E, representing bearing measurements between two vertices. Note that not all robots in the environment are observing a landmark. Each edge in the graph between two robots signifies that relative bearing measurements occur between the two robots and that they can communicate information with each other. Note that the graph in Figure 29 does not have every robot communicating with every other robot, but rather has a sparse set of edges. We assume that all edges in the graph are bi-directional.

Using this basic scenario, we would like each robot to be capable of estimating its navigation state without drift. To more fully describe this system, we will first describe the kinematic model for each robot, followed by the information communicated and used for navigation state estimation.

The navigation state for the ith robot is

$$X_i = [x_i, y_i, \psi_i]^T, \tag{124}$$

where x_i and y_i are the Cartesian coordinates of the robot with respect to a world coordinate frame, and ψ_i is the current heading of the robots.

The kinematics for each robot is given by

$$\dot{X}_i = \begin{bmatrix} V_i \cos(\psi_i) \\ V_i \sin(\psi_i) \\ \omega_i \end{bmatrix} \tag{125}$$

where V_i and ω_i are the robot velocity and turn rate, respectively. We assume $V_i \geq 0$ and $0 \leq \psi_i \leq 2\pi$.

The exteroceptive measurements obtained by each robot are the bearing of landmarks and other robots as measured by a camera. The bearing angle measured between robot i and object j is given by

$$\eta_{ij} = \tan^{-1} \frac{y_j - y_i}{x_j - x_i} - \psi_i + v_{ij}, \tag{126}$$

where v_{ij} is normally distributed, white noise with standard deviation σ_v. If the i_{th} robot has n_i robots and m_i landmarks in its field of view, then the measurement vector for i_{th} robot is given by

$$Z_i = [\eta_{i,1}, \eta_{i,2}, \cdots \eta_{i,k}], \tag{127}$$

where $k = n_i + m_i$.

With the measurement and state vector for each robot defined, we can define the system state as

$$X = [X_1^T, X_2^T, \cdots, X_N^T]^T, \tag{128}$$

where $X \in R^{3N}$, and the total measurement vector as

$$Z = [Z_1^T, Z_2^T, \cdots, Z_N^T]^T, \tag{129}$$

where $Z \in R^M$ and $M = \sum_{i=1}^{N}(n_I + m_i)$.

For state estimation, an Extended Kalman Filter (EKF) is used with the state vector defined in Equation (128) and the measurement vector defined in Equation (129). The full equations for the EKF are:

$$\hat{X}(k|k-1) = \hat{X}(k-1|k-1) + \dot{X}(k-1|k-1)T_s \tag{130}$$
$$P(k|k-1) = F_k P(k-1|k-1)F_k^T + Q_k \tag{131}$$
$$K_k = P(k|k-1)H_k^T(H_k P(k|k-1)H_k^T + R_k)^{-1} \tag{132}$$
$$\hat{X}(k|k) = \hat{X}(k|k-1) + K_k[Z_k - h(\hat{X}(k|k-1))] \tag{133}$$
$$P(k|k) = (I - K_k H_k)P(k|k-1), \tag{134}$$

where $\hat{X}(k|k-1)$ represents the estimate at time k given the measurements up through time $k-1$, $F_k = I_{3n\times 3n} + T_s \frac{\partial \dot{X}_i}{\partial X_i}$ is the transition matrix of the kinematic equations (125), $h(\cdot)$ is the measurement equation expressed in (126), H_k is the Jacobian of that equation, and T_s is the sample time.

B. *Observability Analysis*

A system is observable if its states can be distinguished using a finite sequence of measurements. Observability analysis is a well-established method for determining whether the information available from measurements is sufficient to estimate the states[116]. If the entire state is not observable, the observability analysis can also be used to determine which states or linear combination of states are unobservable. Therefore, we undertake an observability analysis of the cooperative navigation system described in Section VIII-A to determine if it is possible to enable drift-free navigation. If the system is fully observable, it means that the estimation error can be limited by the input measurement noises and will not increase with time.

In the two sections that follow, we briefly review how observability is computed for non-linear systems and introduce some terminology used to describe the connections in our cooperative navigation system.

1) Computing observability for a non-linear system: Because the system is non-linear (e.g., bearing-only measurements and robot kinemetics) we cannot apply traditional linear techniques for evaluating observability. Instead, we analyze local observability in a non-linear system as introduced in [117]. The observability of nonlinear systems can be determined using Lie-derivatives. Let

$$l(x) = \begin{bmatrix} L_f^0(h) \\ \cdots \\ L_f^{n-1}(h) \end{bmatrix} \tag{135}$$

be a column vector of Lie-derivatives where

$$\begin{aligned} L_f^0(h(x)) &= h(x) \\ L_f^1(h(x)) &= \left\langle \frac{\partial}{\partial x}[L_f^0(h(x))], \dot{X} \right\rangle \\ &\cdots \\ L_f^{n-1}(h(x)) &= \left\langle \frac{\partial}{\partial x}[L_f^{n-2}(h(x))], \dot{X} \right\rangle. \end{aligned}$$

Note that the $< a, b >$ operator denotes the inner product of a and b, and that we are assuming $h(x)$ produces a single measurement. If more than one measurement is available, each measurement will add its own set of rows to a concatenated observability matrix.

The observability matrix $\mathcal{O}$ using Lie-derivatives is computed as

$$\mathcal{O}(x) = \frac{\partial l(x)}{\partial x}. \tag{136}$$

A system is locally observable at x if the observability matrix is full rank at x. If $\mathcal{O}(x)$ is not full rank, the row space of the matrix gives the locally observable modes at x, while the locally unobservable modes at x are in the null space of $\mathcal{O}$.

2) Observability analysis for an n-node RPMG: In this section our objective is two determine the maximum achievable rank of the nonlinear observability matrix for a RPMG with n nodes. We show that the maximum rank is $3(n-1)$. Furthermore, we derive necessary and sufficient conditions for achieving rank $3(n-1)$ in terms of connectivity of the graph and vehicle motion. To derive these conditions we first compute the observability of a 2-agent system with bearing-only measurements between the agents. From this analysis, we state and prove four lemmas, leading to the final theorem of this section on observability of cooperative navigation systems without landmarks.

Observability analysis for a 2-node system

Let η_{ij} be the bearing measurement between two vehicles measured by the ith node (the sensor is physically located on vehicle i). Vehicle motion is governed by Equation (125). The combined state vector to be estimated is $X = [X_i \; X_j]^T$ and $\dot{X} = [f_i, \; \omega_i, \; f_j, \; \omega_j]^T$, where, $f_i = [V_i \cos\psi_i \; V_i \sin\psi_i]^T$. Let us now compute the non-linear observability matrix for this two-node system.

Zeroth order Lie-derivative ($L^0_{f(h)}$): From the definition of Lie-derivatives we have,

$$L^0_{f(h)} = \eta_{ij}$$

First order Lie-derivative ($L^1_{f(h)}$):

Differentiating $L^0_{f(h)}$ we obtain,

$$\nabla L^0_{f(h)} = \begin{bmatrix} H^T_{1ij} & -1 & -H^T_{1ij} & 0 \end{bmatrix}$$

where,

$$\begin{aligned} H^T_{1ij} &= \begin{bmatrix} -a_{ij} & b_{ij} \end{bmatrix}, \\ a_{ij} &= \frac{y_i - y_j}{R^2_{ij}}, \\ b_{ij} &= \frac{x_i - x_j}{R^2_{ij}}, \\ R^2_{ij} &= (x_i - x_j)^2 + (y_i - y_j)^2. \end{aligned} \tag{137}$$

By definition,

$$\begin{aligned} L^1_{f(h)} &= \nabla L^0_{f(h)} \cdot \dot{X} \\ &= H^T_{1ij}(f_i - f_j) - \omega_1. \end{aligned}$$

Second order Lie-derivative ($L^2_{f(h)}$): Differentiating $L^1_{f(h)}$ we obtain

$$\nabla L^1_{f(h)} = \begin{bmatrix} H^T_{2ij} & H^T_{1ij}F_i & -H^T_{2ij} & -H^T_{1ij}F_j \end{bmatrix}$$

where,

$$\begin{aligned} H^T_{2ij} &= (f_i - f_j)^T J_{1ij}, \\ J_{1ij} &= \frac{\partial H_{ij}}{\partial X_i} = \begin{bmatrix} 2a_{ij}b_{ij} & (a^2_{ij} - b^2_{ij}) \\ (a^2_{ij} - b^2_{ij}) & -2a_{ij}b_{ij} \end{bmatrix}, \\ F_i &= \frac{\partial f_i}{\partial \psi_i} = \begin{bmatrix} -V_i \sin\psi_i \\ V_i \cos\psi_i \end{bmatrix}. \end{aligned} \tag{138}$$

By definition we have,

$$\begin{aligned} L^2_{f(h)} &= \nabla L^1_{f(h)} \cdot \dot{X} \\ &= (f_i - f_j)^T J_{1ij}(f_i - f_j) + H^T_{ij}(F_i\omega_i - F_j\omega_j) \end{aligned}$$

Differentiating $L^2_{f(h)}$, we obtain

$$\nabla L^2_{f(h)} = \begin{bmatrix} H^T_{3ij} & H^T_{4ij} & -H^T_{3ij} & H^T_{5ij} \end{bmatrix}$$

where

$$\begin{aligned} H^T_{3ij} &= (f_i - f_j)^T J_{2ij} + (F_i\omega_i - F_j\omega_j)^T J_{1ij}, \\ J_{2ij} &= 2(a^2 + b^2)[J^a_{2ij} J^b_{2ij}](f_i - f_j), \\ J^a_{2ij} &= \begin{bmatrix} a & b \\ b & -a \end{bmatrix}, \\ J^b_{2ij} &= \begin{bmatrix} -b & a \\ a & b \end{bmatrix}, \end{aligned}$$

$$\begin{aligned} H_{4ij}^T &= (f_i - f_j)^T J_{1ij}^T F_i + H_{2ij}^T F_i - \omega_i H_{1ij}^T f_i, \\ H_{5ij}^T &= -(f_i - f_j)^T J_{1ij}^T F_j - H_{2ij}^T F_j + \omega_j H_{1ij}^T f_j. \end{aligned}$$

Higher order Lie-derivatives will be linear combinations of $(f_i - f_j)$ and $(F_i\omega_i - F_j\omega_j)$ and will therefore not contribute to the rank of the observability matrix.

From this information we can write the observability matrix for two vehicles with bearing measurement η_{ij}^k as,

$$\begin{aligned} \mathcal{O}^{ij} &= \begin{bmatrix} \nabla L_{f(h)}^0 \\ \nabla L_{f(h)}^1 \\ \nabla L_{f(h)}^2 \end{bmatrix} \\ &= \begin{bmatrix} H_{1ij}^T & -1 & -H_{1ij}^T & 0 \\ H_{2ij}^T & H_{1ij}^T F_i & -H_{2ij}^T & -H_{1ij}^T F_j \\ H_{3ij}^T & H_{4ij}^T & -H_{3ij}^T & H_{5ij}^T \end{bmatrix} \end{aligned} \tag{139}$$

Lemma 1 (Two nodes): The observability matrix given in Equation (139) has rank three if and only if the following conditions are satisfied.

1) $V_j > 0$,
2) $\dot{X}_i \neq \dot{X}_j$.
3) $\eta_{ij} \neq 0$ *or* $\dot{\eta}_{ij} \neq 0$.

Proof: To prove this lemma we will first prove the sufficiency of the listed conditions, followed by a proof of necessity. When the lemma's conditions are satisfied, the reduced-row echelon form (RREF) of the observability matrix in Equation (139) is:

$$RREF(\mathcal{O}^{ij}) = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 & y_i - y_j \\ 0 & 1 & 0 & 0 & -1 & x_j - x_i \\ 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix},$$

which is obviously a rank 3 matrix.

To prove the necessity of the lemma's conditions, we compute the observability matrix's rank for the case when the conditions are violated. First, to prove the necessity of $V_j \neq 0$, we compute the observability matrix when $V_i > 0$ and $V_j = 0$.

1) Assume that $V_i > 0$ and $V_j = 0$. Plugging these into 139 we obtain,

$$\mathcal{O}^{ij} = \begin{bmatrix} H_{1ij}^T & -1 & -H_{1ij}^T & 0 \\ f_i^T J_{1ij} & H_{1ij}^T F_i & -f_i^T J_{1ij} & 0 \\ f_i^T J_{2ij} & 2 f_i^T J_{1ij}^T F_i & -f_i^T J_{2ij} & 0 \end{bmatrix},$$

and $rank(\mathcal{O}^{ij}) = 2 < 3$.

2) Plugging $V_i = 0$ and $V_j = 0$ in (139) we,have

$$\mathcal{O}^{ij} = \begin{bmatrix} H_{1ij}^T & -1 & -H_{1ij}^T & 0 \\ 0_{1\times 2} & 0 & 0_{1\times 2} & 0 \\ 0_{1\times 2} & 0 & 0_{1\times 2} & 0 \end{bmatrix}$$

Hence, $rank(\mathcal{O}^{ij}) = 1 < 3$.

This proves the necessity of $V_j > 0$.

If $\dot{X}_i = \dot{X}_j$, then $f_i = f_j$ and $\omega_i = \omega_j$, leading to an observability matrix of

$$\begin{aligned} \mathcal{O}^{ij} &= \begin{bmatrix} H_{1ij}^T & -1 & -H_{1ij}^T & 0 \\ 0_{1\times 2} & H_{1ij}^T F_i & 0_{1\times 2} & -H_{1ij}^T F_i \\ 0_{1\times 2} & \alpha & 0_{1\times 2} & -\alpha \end{bmatrix} \\ \alpha &= H_{2ij}^T F_i - \omega_i H_{1ij}^T f_i. \end{aligned}$$

The rank of this matrix is 2, proving the necessity of $\dot{X}_i \neq \dot{X}_j$.

Finally assume that $\eta_{ij} = 0,\ and\ \dot{\eta}_{ij} = 0$. This condition implies that $\dot{\eta}_{ij}$ and higher derivatives are zero. Therefore, $rank(\mathcal{O}^{ij}) = 1 \leq 3$. ■

Remark 1: It can be verified that the three different sets of measurements η_{ij}, η_{ji} and $[\eta_{ij}, \eta_{ji}]$ all have the same observability rank and observable modes, assuming $V_i, V_j > 0$. In the case that $V_i = 0$, η_{ij} and $[\eta_{ij}, \eta_{ji}]$ will both lead to the same rank 3 observability matrix, while η_{ji} leads to a rank 2 observability matrix as shown above. If both V_i and V_j are 0, then no set of bearing-only measurements between the robots will create an observability matrix of rank 3.

Having determined the necessary and sufficient conditions for obtaining three linearly independent observability vectors between two nodes, we proceed to introduce a *proper RPMG*.

Definition 1: A *proper RPMG* is an ordered pair (X, E') consisting of a set of vertices X (the same set of vertices as the original, physical RPMG), and a set of edges E'. The set E' is a subset of E from the original RPMG and is defined as:

$$E' = \{e : e \in E \text{ and } e = (i, j), rank(\mathcal{O}^{ij}) = 3\}$$

The creation of a proper RPMG has two effects:

1) The proper RPMG simplifies further derivation of conditions required for drift-free navigation. Rather than repeatedly restating the conditions in Lemma 1, we enforce the conditions by pruning edges from the physical RPMG to create the proper RPMG.
2) The proper RMPG is the first step in creating a useful link between the system observability matrix and the graph structure of the system. We show in the following sections that by analyzing the proper RPMG, the overall observability of the system can be determined.

3) Observability of multi-agent systems:

Lemma 2 (Similarity of 3-node configurations): There are four possible connected graphs for three nodes as shown in Figure 30. Assuming these configurations are in a proper RPMG, all four of these configurations yield a set of vectors in the observability-matrix that span the same space. Therefore, all four of the configurations lead to the same rank of the observability matrix.

Proof: The reduced row echelon form (RREF) of the

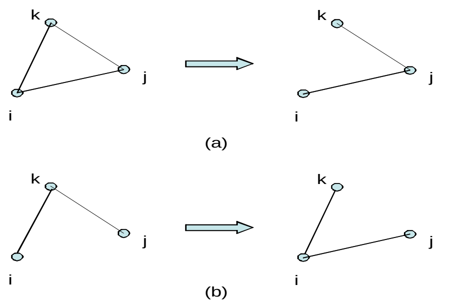


Fig. 30. The observability conditions between these four possible configurations of a connected, 3-vertex RPMG are identical. Of particular note, we can perform redundant edge removal (subfigure (a)) or a side exchange (subfigure (b)) and not change the observability of the system.

observability matrix for each connected 3-node graph is

$$RREF(\mathcal{O}^{ijk}) = \left[I_{6\times 6} : \begin{array}{ccc} -1 & 0 & y_i - y_k \\ 0 & -1 & x_k - x_i \\ 0 & 0 & -1 \\ -1 & 0 & y_j - y_k \\ 0 & -1 & x_k - x_j \\ 0 & 0 & -1 \end{array} \right].$$

Because the RREF form of the four observability matrices are the same, each 3-node connected graph introduces observability matrix vectors that span the same space. ■

To extend the observability analysis to arbitrarily large graphs, we block partition the observability matrix $\mathcal{O}^{ij}$ for each measurement (edge e_{ij}) between vehicle i and j, creating two sub-matrices $\mathcal{O}^{ij}_i$ and $\mathcal{O}^{ij}_j$ corresponding to the columns for agent i and j, respectively. In an n vehicle system, the sub-matrix $\mathcal{O}^{ij}_i$ will be placed in the columns corresponding with agent i, leading to a $3 \times 3n$ matrix of the form:

$$\mathcal{O}^{ij} = \begin{bmatrix} 0_{3\times 3(i-1)} & \mathcal{O}^{ij}_i & 0_{3\times(3(j-1)-3i)} & \mathcal{O}^{ij}_j & 0_{3\times(n-3j)} \end{bmatrix} \tag{140}$$

for each edge in the proper RPMG. The observability matrix for the entire RPMG is derived by concatenating the matrices from all edges in the graph. Using this form of the observability matrix, we can prove several useful properties of arbitrary proper RPMGs.

Lemma 3 (Equivalent Graphs): Given a proper RPMG, any three-node connected sub-graph can be replaced with any other connected three-node sub-graph without modifying the rank of the associated observability matrix.

Proof: Given the block form of the observability matrix shown in Equation (140), this is a simple extension of Lemma 2. ■

Lemma 4 (Two level tree): A proper RPMG with n nodes, consisting of a single node connected directly to all other nodes in the network, with no other edges present in the network (a *two-level tree*, e.g., Figure 31(f)), will have a local observability matrix of rank $3(n-1)$.

Proof: Assume without loss of generality that the root node is labeled $i = 1$. In this case, the system observability matrix can be written in block form as

$$\mathcal{O}^{1,\dots,n} = \begin{bmatrix} \mathcal{O}^{12}_1 & \mathcal{O}^{12}_2 & 0_{3\times3} & \cdots & 0_{3\times3} \\ \mathcal{O}^{13}_1 & 0_{3\times3} & \mathcal{O}^{13}_3 & \cdots & 0_{3\times3} \\ \vdots & \vdots & \cdots & \ddots & \vdots \\ \mathcal{O}^{1n}_1 & 0_{3\times3} & \cdots & 0_{3\times3} & \mathcal{O}^{1n}_n \end{bmatrix}. \tag{141}$$

Because $\mathcal{O}^{1j}_j$ is the only non-zero entry in the jth block column, its contribution to the observability matrix rank will be linearly independent of the contribution of $\mathcal{O}^{1k}_k$ for any $k \neq j$. From Lemma 1, if $V_i \neq 0$ and $\dot{X}_i \neq \dot{X}_j$, then each $\mathcal{O}^{1j}$ block row will have three linearly independent rows. Because there are $n-1$ edges in a two-level tree, the local observability matrix will have $3(n-1)$ linearly independent rows. ■

Theorem 1: Any connected, proper RPMG of n nodes will have an observability matrix rank of $3(n-1)$.

Proof: Using Lemma 4, this theorem can be restated as: "Any connected, proper RPMG has the same local observability matrix rank as a two-level tree." To prove equivalence, we present an algorithm that converts any connected proper RPMG to a two-level tree. Every step in this algorithm preserves rank of the observability matrix, proving the rank-equivalence between any connected proper RPMG and a two-level tree. A pictorial representation of this algorithm is also shown in Figure 31.

Algorithm:

1) Randomly choose a node, i that will become the root node of the two-level tree.
2) Compute the minimum distance of all nodes in the graph to node i.
3) Place all non-root nodes nodes in one of two sets: (1) nodes that are minimum distance 1 away from node i (S_1), and (2) nodes that are more than minimum distance 1 away from node i (S_2).
4) Select a node (j) in S_2 that is distance 2 away from node i. Select a node (k) in set S_1 that connects to both i to j. Note that these three nodes (i, j, k) from a three-node connected sub-graph.
5) Replace the three-node connected sub-graph with a three-node connected sub-graph where both nodes j and k are distance 1 away from node i (a *side exchange* step). Recompute the distance of all nodes from i and re-form sets S_2 and S_1. Using Lemma 3, this new graph is rank-equivalent to the original graph.
6) Repeat steps 4 and 5 until $S_2 = \emptyset$. Note that with each execution of these two steps, a single node is removed from S_2 and added to S_1. The minimum distance of other nodes in S_2 may also decrease. Because the graph is always connected, all nodes will eventually move to S_1 as this algorithm executes.
7) Select an edge (e_{jk}) between two nodes (j, k) in S_1. Form a three-node connected sub-graph between nodes i, j, and k. Replace with a sub-graph that connects i to j and i to k, but removes e_{jk} (a *redundant edge*

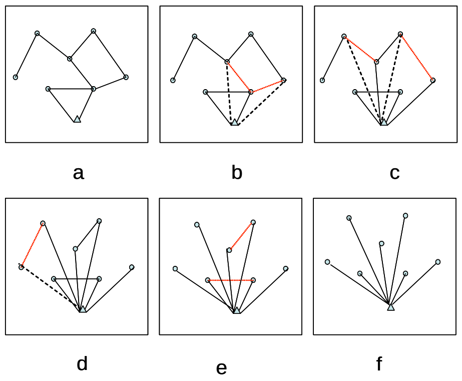


Fig. 31. An example of converting a connected RPMG to a two-level tree as described in Theorem 1. The subfigures correspond with: (a) the original RPMG, (b) Side exchanges to bring two nodes distance 2 from the root node to distance 1 (step 5), (c) two more side exchanges, (d) the final side exchange, resulting in a graph where every "non-root" node is distance 1 away from the root, (e) the removal of any redundant edges (step 7) and (f) the final, two-level tree.

removal step). From Lemma 3, this modification does not affect the observability matrix rank.

8) Repeat step 7 until no edges are found between nodes in S_1.

■

Corollary 1: If a proper RPMG contains m disjoint subgraphs and n nodes, the total observability matrix rank will be $3n - 3m$.

C. Observability analysis of n-node RPMG with known landmarks

Up to this point we have assumed that the only information available to the agents performing navigation are interoceptive sensor measurements and measurements of other agents in the system. The maximum rank of the observability matrix with these restrictions is $3(n-1)$. In other words, while the relative location and orientation of every agent with respect to all other agents can be observed, the global location and orientation is unobservable. This unobservability of the global position and orientation will cause the system to drift over time.

To enable drift-free navigation, more measurements must be available to the system to obtain a full-rank observability matrix. In this section, we show that by adding a minimal number of landmarks with fixed location, the entire system becomes observable, enabling drift-free navigation. We begin by deriving the observability matrix vectors obtained by a single agent obtaining bearing-only measurements of a landmark. We then derive necessary and sufficient conditions for a fully observable system.

1) Vehicle-Landmark observability: Assume that vehicle i measures bearing η_{ip} from a known landmark p. Note that in this case, we are only concerned with the navigation state of the agent as the landmark location is already known. The Lie-derivatives of this observation follow.

Zeroth order Lie-derivative:

$$L^0_{f(h)} = \eta_{ip}$$

First order Lie-derivative: Differentiating $L^0_{f(h)}$ we obtain,

$$\nabla L^0_{f(h)} = \begin{bmatrix} H^T_{1ip} & -1 \end{bmatrix}$$

where, H^T_{1ip} is defined in (137).

By definition,

$$\begin{aligned} L^1_{f(h)} &= \nabla L^0_{f(h)} \cdot \dot{X} \\ &= H^T_{1ip} f_i - \omega_i \end{aligned}$$

Second and higher order Lie-derivatives will be multiples of f_i and F_i. Therefore, the maximum contribution of any one landmark measurement to the observability matrix rank is 2.

Differentiating $L^1_{f(h)}$ we obtain

$$\nabla L^1_{f(h)} = \begin{bmatrix} H^T_{2ip} & H^T_{1ip} F_i \end{bmatrix}$$

where,

$$H^T_{2ip} = f_i^T J_{1ip}$$

and J_{1ip} is defined in (138). Therefore, the observability matrix of bearing measurements between a vehicle and landmark can be written as,

$$\begin{aligned} \mathcal{O}^{ip}_i &= \begin{bmatrix} \nabla L^0_{f(h)} \\ \nabla L^1_{f(h)} \end{bmatrix} && (142) \\ &= \begin{bmatrix} H^T_{1ip} & -1 \\ H^T_{2ip} & H^T_{1ip} F_i \end{bmatrix}. && (143) \end{aligned}$$

In reduced row echelon form, we obtain:

$$RREF(O^{ip}_i) = \begin{bmatrix} 1 & 0 & y_i - y_p \\ 0 & 1 & x_p - x_i \end{bmatrix}. \quad (144)$$

Remark 2: The number of linearly independent rows achieved by a single landmark observation will be two if and only if following conditions are satisfied:

1) $V_i > 0$.
2) $\eta_{ip} \neq 0$ *or* $\dot{\eta}_{ip} \neq 0$

2) Landmark observability with multiple agents: We now extend the one vehicle observability of landmarks to multi-vehicle systems. Note that observations of a landmark by agent i can be added to the observability matrix using the block form

$$\mathcal{O}^{ip} = [0_{3\times 3(i-1)}) \; \mathcal{O}^{ip}_i \; 0_{3\times 3(n-i)})]^T$$

Remark 3: Due to the block-matrix form of the system observability matrix, the observability matrix vectors due to observing landmarks and those due to observing other agents are always linearly independent.

Lemma 5: The number of linearly independent observability rows due to landmark observations in a connected, proper RPMG is equal to

$$\min\left(3, \sum_{p\in L} \min\left(2, \sum_{i\in X} rank(\mathcal{O}^{ip})\right)\right),$$

where L is the set of landmarks observed by any node within the graph.

Proof: Assume a 2-agent network with one agent-to-agent measurement and one landmark p. The observability matrix for this network can be written as

$$\mathcal{O} \;=\; \begin{pmatrix} \mathcal{O}_i^{ij} & \mathcal{O}_j^{ij} \\ \mathcal{O}_i^{ip} & 0_{2\times 3} \end{pmatrix}. \qquad (145)$$

Finding the reduced row echelon form of this matrix, we obtain:

$$RREF(\mathcal{O}) = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 & y_i - y_j \\ 0 & 1 & 0 & 0 & -1 & x_j - x_i \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & y_j - y_p \\ 0 & 0 & 0 & 0 & 1 & x_p - x_j \end{bmatrix}. \qquad (146)$$

Note that this is equivalent to

$$RREF(\mathcal{O}) = \begin{bmatrix} \mathcal{O}_i^{ij} & \mathcal{O}_j^{ij} \\ 0_{2\times 3} & \mathcal{O}_j^{jp} \end{bmatrix}. \qquad (147)$$

This rank-equivalent matrix transformation demonstrates two important points:

- The measurement of a landmark by an agent is equivalent, in terms of observability, to any other agent observing the same landmark. Because the maximum number of linearly independent rows that can be achieved from observing a landmark is two, the number of linearly independent rows in the observability matrix due to a single landmark will be:

$$\min\left(2, \sum_{i\in X} rank(\mathcal{O}^{ip})\right).$$

- If an agent observes another landmark q, the measurement of q causes linearly independent rows to be added to the observability matrix if and only if the location of q is different from the location of p. Therefore, the observation of additional landmarks increases the number of linearly independent rows in the observability matrix in an additive manner, leading to

$$\sum_{p\in L} \min\left(2, \sum_{i\in X} rank(\mathcal{O}^{ip})\right)$$

linearly independent rows in the observability matrix.

Combining these two observations with the fact that (a) the maximum rank of the observability matrix is $3n$ and (b) the observability matrix of a connected proper RPMG already has $3(n-1)$ linearly independent rows, the number of linearly independent rows added to the observability matrix with the observation of known-location landmarks will be:

$$\min\left(3, \sum_{p\in L} \min\left(2, \sum_{i\in X} rank(\mathcal{O}^{ip})\right)\right),$$

■

Corollary 2: Using Lemma 5, we can derive three basic conditions under which the local observability matrix rank is increased by three. These conditions are:

- Observing three landmarks anywhere within the system, irrespective of agent movement. As each landmark observation adds at least one linearly independent row to the observability matrix, three landmarks is always sufficient for full observability of a connected, proper RPMG.
- Observing two landmarks anywhere within the system, with one landmark observed by at least two agents, irrespective of agent motion. Any one landmark can provide up to two linearly independent rows to the observability matrix. Even if agents are not moving, having two agents observe the same landmark will still lead to two linearly independent rows from the one landmark. Adding one other landmark yield three linearly independent rows in the observability matrix.
- Observing two landmarks, with at least one observing agent moving ($V > 0$) *not* directly at the landmark it is observing. From Lemma 2, if an agent is moving, two linearly independent rows are obtained from observing a single landmark. By adding one more landmark, three linearly independent rows are obtained.

We now state the main theorem of the section:

Theorem 2 (Drift-free Navigation): Consider a RPMG with n agent nodes and m different landmarks. This system is completely observable (i.e, $rank(\mathcal{O}) = 3n$) if the proper RPMG is connected and one of the following conditions are true: (1) $m \geq 3$, (2) $m = 2$ and at least one landmark is observed by more than one agent, or (3) $m = 2$ and one of the observing agents is moving.

Proof: This theorem is proved by combining Theorem 1 and Lemma 5. In any of the scenarios discussed in this theorem, the rank of the observability matrix will be $3n$. ■

Corollary 3: In a proper RPMG with disjoint sub-graphs, drift-free navigation is still achieved when each sub-graph meets one of the conditions in Theorem 2.

D. Simulation Results

To validate the theorems developed in previous sections, we simulated cooperative navigation with a group of ten vehicles and two landmarks. We assume that only two agents have access to the landmark positions. Every vehicle has a sensor from which it can measure bearing to other vehicles and landmarks within its sensor range R_{sensor}.

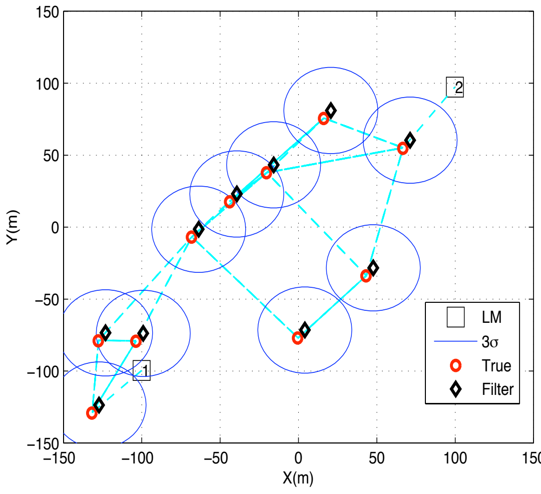


Fig. 32. Initial RPMG with 10 vehicle nodes and two known landmarks. Only vehicle one and two have the access to the landmark location. The circles around each node represent the initial 3σ uncertainty, the red circle represents the true position of the agents, and the black diamonds represent the estimated position.

We also assume that the graph is always connected. Initial position and heading uncertainty on each vehicle is $[P_{x0}\ P_{y0}\ P_{\psi_0}] = [5m\ 5m\ 0.1rad]$. Each vehicle velocity is $V = 4m/s$. The standard deviation of the noise on the velocity and turn rate measurements for each vehicle is $[\sigma_v\ \sigma_\omega]^T = [0.2m/s\ 0.2rad/s]^T$. The initial RPMG of ten vehicles and two landmarks with initial 3σ uncertainty ($t = 0s$) is shown in Fig. 32. Fig. 33 and 34 shows the RPMG at $t = 10s$ and $t = 40s$ respectively. It can be seen that the uncertainty in position estimates of each vehicle decreases with time, verifying the observability of all agent's navigation states. Fig. 35 and 36 shows the navigation state estimation error for vehicle one and vehicle six respectively. Vehicle one has access to the landmark positions whereas vehicle six does not have direct access. Despite this difference, the error in all three states ($[x, y, \psi]$) decrease over time to a constant bound. Note also that these plots show navigation accuracy over a period of 1000s, demonstrating the independence of the navigation error to the amount of time the system has been running.

Contrast these results with the error plots shown in Figure 37. These error plots represent the case when cooperative navigation is used across ten agents, but no fixed landmarks are present in the environment. In contrast to the case with two landmarks available to the system, the error increases with time due to the unobservability of the system.

E. Experimental Results

To demonstrate the feasibility of our drift-free navigation in real-world systems, we prototyped a cooperative navigation system with 3 robots. Each robot (shown in Figure 38) uses wheel encoders to measure turn rate and velocity, an omnidirectional camera to measure other robots and fixed landmarks, and an ASUS EEE PC for on-board control, image processing, and communication. The experimental setup is shown in Fig. 39. It consist of three robots with different colors (green, blue, and orange), simplifying the identification of the robots in other robot's images. The robots communicate using an 802.11 wireless router. The environment is instrumented with an overhead camera to obtain the true navigation states of the robots. Figure 40 shows the trajectory of three robots generated by encoders, the cooperative navigation system (CNS), and truth. It can be seen that the CNS estimates are very close to the true states. In Figure 41, 42, and 43, we show the CNS and

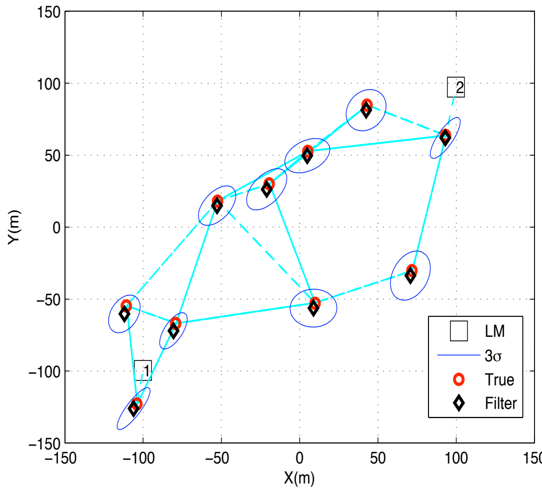


Fig. 33. The RPMG at $t = 10s$. The interpretation of this graph is the same as Figure 32. Note that the uncertainty in navigation estimates has decreased for all nodes compared to Figure 32

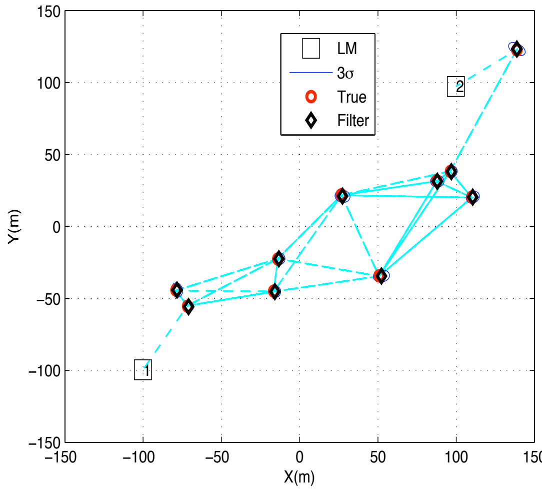


Fig. 34. The RPMG at $t = 50s$. The interpretation of this graph is the same as Figure 32. By this time, the navigation state uncertainties for all nodes have essentially reached their lower bound.

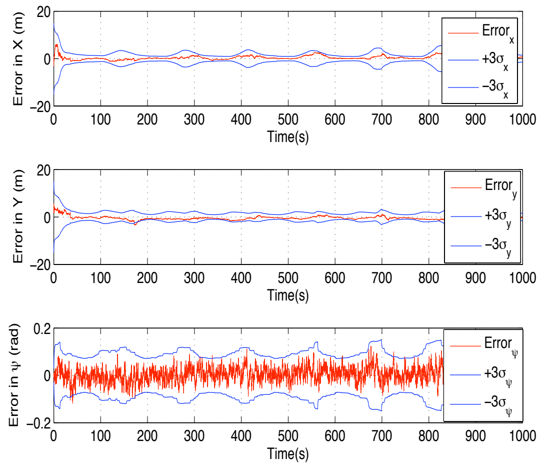


Fig. 35. Navigation state error plots of vehicle 1 (vehicle one has access to landmark positions). The red curve is the error in the CNS state estimate and blue curves represents the $\pm 3\sigma$ bounds. Note that the $3-\sigma$ bounds are not increasing with time.

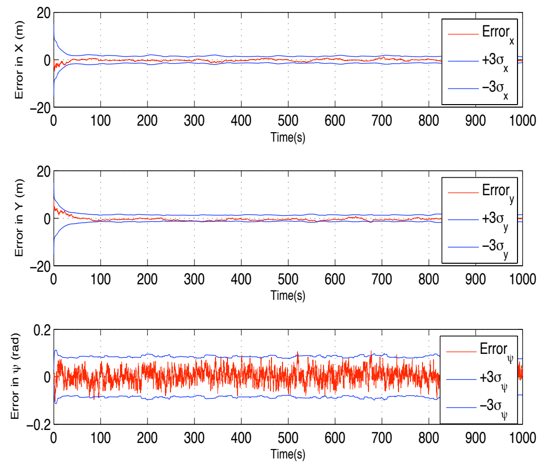


Fig. 36. Navigation state error plots of vehicle 6 (no direct access to landmark positions). The red curve is the error in the navigation state estimate and blue curves represents the $\pm 3\sigma$ bounds. Note that Despite the lack of direct access to a landmark, the error bounds are not increasing with time.

encoder estimation error in X, Y, and ψ, respectively, for the blue robot. We also show the $\pm 3\sigma$ bounds. Note that the estimation error of the CNS is bounded with time (over 600s), while the encoder navigation estimates contain significant drift.

IX. Results of Funding

As a result of the funding provided for this work, the following papers were published:

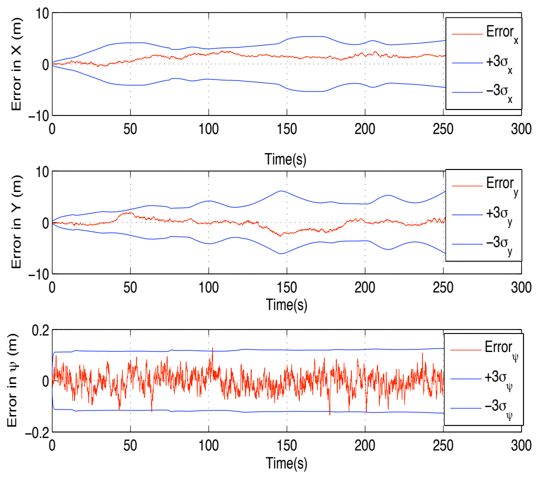


Fig. 37. Navigation state error plots of a vehicle when there are no landmarks in the system. Red curve is the error in the navigation state estimate and blue curves represents the $\pm 3\sigma$ bounds. Note that the error bounds are increasing with time.

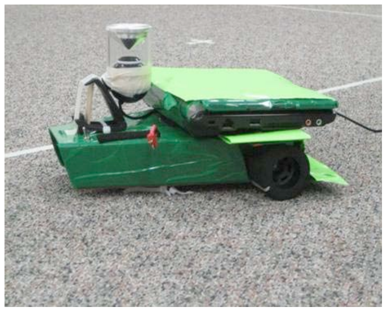


Fig. 38. Stinger robot assembly with serializer, EEE PC and omnidirectional camera

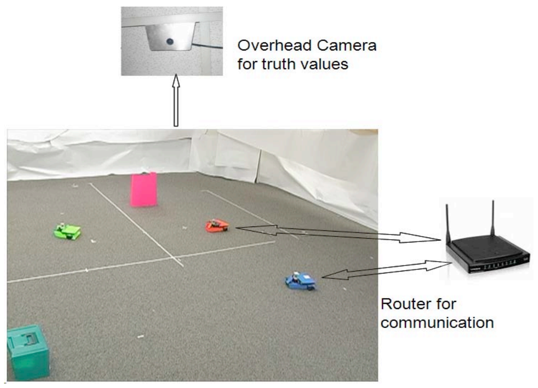


Fig. 39. Experimental setup with three robots. An overhead camera is used to measure truth values. Each agent communicates with the other agents through an 802.11 wireless network.

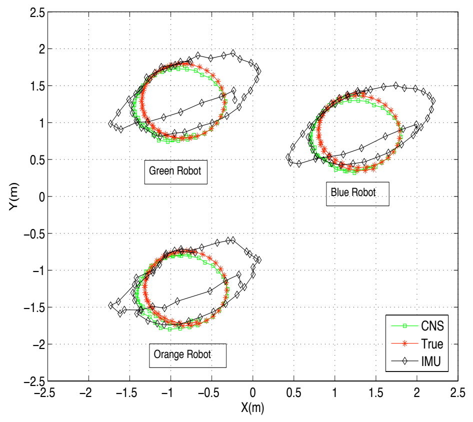


Fig. 40. Trajectory generated by CNS, encoders and overhead camera.

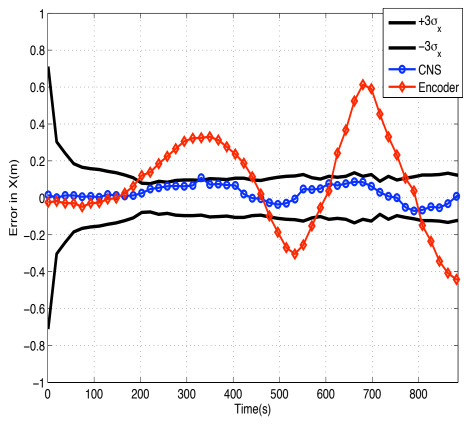


Fig. 41. Error in X (Blue Robot): Error in CNS estimate (blue curve). Error in encoder estimate (red curve). Black curves are $\pm 3\sigma$ bounds.

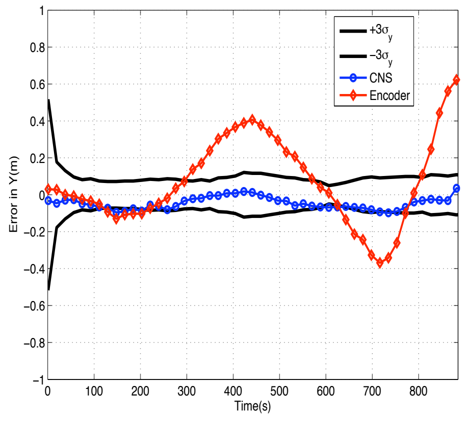


Fig. 42. Error in Y (Blue Robot): Error in CNS estimate (blue curve). Error in encoder estimate (red curve). Black curves are $\pm 3\sigma$ bounds.

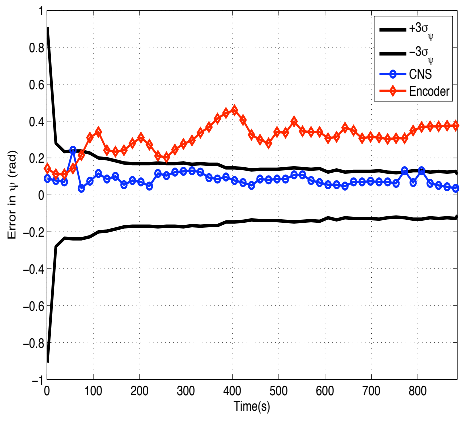


Fig. 43. Error in ψ (Blue Robot): Error in CNS estimate (blue curve). Error in encoder estimate (red curve). Black curves are $\pm 3\sigma$ bounds.

A. Journal Papers

1) "A Comparison of Two Image and Inertial Sensor Fusion Techniques for Navigation in Unmapped Environments," C.N. Taylor, M.J. Veth, J.F. Raquet, M.M. Miller, accepted in IEEE Transactions on Aerospace and Electrical Systems.
2) "Enabling Navigation of MAVs through Inertial, Vision, and Air Pressure Sensor Fusion," C.N. Taylor, Lecture Notes in Electrical Engineering, v. 35, pp. 143-158, 2009.
3) "Inertially Aided Visual Odometry for Miniature Air Vehicles in GPS-denied Environments," B.B. Ready, C.N. Taylor, Journal of Intelligent and Robotic Systems, v. 55, no. 2-3, pp. 203-221, 2009.
4) "Stabilization of Video from Miniature Air Vehicles for Target Localization," D.L. Johansen, J.K. Hall, R.W. Beard, C.N. Taylor, Journal of Aerospace Computing, Information, and Communication, Vol 5, no. 8, pp. 251-273, 2008.

B. Conference Papers

1) "Cooperative GPS Navigation," S. Quebe, J. Campbell, S. DeVilbiss, C.N. Taylor, accepted in Proceedings, IEEE Precision Location and Navigation Symposium, 2010.
2) "Creation of Geo-Referenced Mosaics from MAV Video and Telemetry using Constrained Optimization and Bundle Adjustment," B. Heiner and C.N. Taylor, in Proceedings, IEEE International Conference on Intelligent Robots and Systems, 2009.
3) "Reactive Collision Avoidance for Fixed-wing MAVs Flying in Urban Terrain," R. Sharma, J. Saunders, C.N. Taylor, and R.W. Beard, in Proceedings, AIAA Conference on Guidance, Navigation, and Control, 2009.
4) "Improved MAV Attitude Estimation Through Coupled Acceleration Estimation," B. B. Ready and C. N.

Taylor, in Proceedings, Institute of Navigation, International Technical Meeting, 2009.

5) "Long-term Accuracy of Camera and IMU Fusion-based Navigation Systems," C. N. Taylor, in Proceedings, Institute of Navigation, International Technical Meeting, 2009.
6) "Vision-based Distributed Cooperative Navigation for MAVs in GPS Denied Areas," R. Sharma and C. N. Taylor, AIAA Infotech@Aerospace, 2009.
7) "Payload Directed Flight of Micro Air Vehicles," R.W. Beard, C.N. Taylor, J. Saunders, R. Holt, T.W. McLain, AIAA Infotech@Aerospace 2009.
8) "An Automatic System for Creating Geo-referenced Mosaics from MAV Video," C.N. Taylor and E.D. Andersen, in Proceedings, IEEE International Conference on Intelligent Robots and Systems, Nice, France, Sept., 2008
9) "Fusion of Inertial, Vision, and Air Pressure Sensors for MAV Navigation," C.N. Taylor, in Proceedings, IEEE International Conference on Multi-sensor Fusion and Integration for Intelligent Systems, Seoul, Korea, Aug., 2008.
10) "Cooperative Navigation of MAVs in GPS-Denied Areas," R. Sharma, C.N. Taylor, in Proceedings, IEEE International Conference on Multi-sensor Fusion and Integration for Intelligent Systems, Seoul, Korea, Aug., 2008.
11) "Onboard System for Synchronizing Video and Telemetry on a Small UAV," A. Rodriguez, C. Taylor, Y. Aregawi, R. Dennis, and T. Jenkins, SPIE Defense, Security, and Sensing Conference, Orlando, FL, March 2008.
12) "Improving MAV Pose Estimation Using Visual Information," E.D. Andersen and C.N. Taylor, published in Proceedings, IEEE International Conference on Intelligent Robots and Systems, San Diego, CA, Oct-Nov, 2007.
13) "Wind Estimation Using an Optical Flow Sensor on a Miniature Air Vehicle," A. Rodriguez, E. Andersen, J. Bradley, and C.N. Taylor, published in Proceedings, AIAA Conference on Guidance, Navigation, and Control, Hilton Head, SC, Aug 2007.
14) "Particle Filter Based Mosaicking for Tracking Forest Fires," J. Bradley and C.N. Taylor, published in Proceedings, AIAA Conference on Guidance, Navigation, and Control, Hilton Head, SC, Aug 2007.
15) "Improving Accuracy of MAV Pose Estimation using Visual Odometry," B.B. Ready and C.N. Taylor, published in Proceedings, 2007 American Control Conference, pp.3721-3726.

C. Theses

1) "A Surveillance System to Create and Distribute Geo-referenced Mosaics from SUAV Video," Evan Andersen, M.S. Thesis, Brigham Young University, 2008. Available at http://contentdm.lib.byu.edu/ETD/image/etd2416.pdf.
2) "Real-time Wind Estimation and Video Compression On-board Miniature Aerial Vehicles," Andres Rodriguez-Perez, M.S. Thesis, Brigham Young University, 2009. Available at http://contentdm.lib.byu.edu/ETD/image/etd2792.pdf.
3) "Construction of Large Geo-referenced Mosaics from MAV Video and Telemetry Data," Benjamin Heiner, M.S. Thesis, Brigham Young University, 2009. Available at http://contentdm.lib.byu.edu/ETD/image/etd3045.pdf.

D. Students Funded

In addition to the publications listed above, the following students each received full or partial funding from this contract:

- Evan Andersen, M.S. student (graduated April 2008).
- Andres Rodriguez-Perez, M.S. student (graduated April 2009).
- Benjamin Heiner, M.S. student (graduated June 2009).
- Bryce Ready, PhD student (estimated graduation, April 2011).
- Rajnikant Sharma, PhD student (estimated graduation, December 2011).

X. Conclusion and Future Work

Over the three-year course of this contract, significant advances were made in (1) the fusion of inertial and visual sensors using frame-to-frame tracking techniques, (2) performing detailed comparisons between frame-to-frame tracking and SLAM techniques, and (3) enabling multiple-agent navigation techniques. These advances results in several published journal and conference papers, and two masters' thesis were completed on this project. Two PhD students (Bryce Ready and Rajnikant Sharma), while not yet finished with their PhDs, will soon be finished and were primarily supported by this research contract.

In the future, the work on multi-agent navigation holds significant promise for enabling complex missions in GPS-denied areas. By utilizing multiple agents, missions of unbounded length can be performed without regard to navigation accuracy. Therefore, techniques that enable practical utilization of multi-agent navigation must be developed. In particular, multi-agent systems that utilize control algorithms to maintain the necessary connectivity for drift-free navigation need to be developed. In addition, while the navigation error is bounded, an examination of what those bounds are needs to be undertaken. With these future advances in multi-agent navigation systems, significant improvements in Air Force capabilities without GPS can be achieved.

References

[1] R. Beard, D. Kingston, M. Quigley, D. Snyder, R. Christiansen, W. Johnson, T. McLain, and M. Goodrich, "Autonomous Vehicle Technologies for Small Fixed Wing UAVs," *AIAA Journal of Aerospace Computing, Information, and Communication*, vol. 2, no. 1, Jan 2005.

[2] D. B. Kingston and R. W. Beard, "Real-Time Attitude and Position Estimation for Small UAV's Using Low-Cost Sensors," in *AIAA Unmanned Unlimited Systems Conference and Workshop*, Chicago, IL, Sept 2004.

[3] "Procerus technologies www.procerusuav.com," 2006. [Online]. Available: http://www.procerusuav.com/products.php

[4] "Micropilot, www.micropilot.com," 2006. [Online]. Available: http://www.micropilot.com/index.htm

[5] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1, 27 June-2 July 2004, pp. I–652–I–659Vol.1.

[6] T. Kanade, O. Amidi, and Q. Ke, "Real-time and 3d vision for autonomous small and micro air vehicles," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 2, 14-17 Dec. 2004, pp. 1655–1662Vol.2.

[7] A. Brown and D. Sullivan, "Inertial navigation electro-optical aiding during GPS dropouts," in *Proceedings of the Joint Navigation Conference*, 2002.

[8] M. J. Veth and J. F. Raquet, "Two-dimensional stochastic projections for tight integration of optical and inertial sensors for navigation," in *National Technical Meeting Proceedings of the Institute of Navigation*, 2006, pp. 587–596.

[9] M. J. Veth, J. F. Raquet, and M. Pachter, "Stochastic constraints for robust image correspondence search," *IEEE Transactions on Aerospace and Electronic Systems*, vol. -, pp. –, 2007, publication forthcoming.

[10] B. B. Ready and C. N. Taylor, "Improving accuracy of mav pose estimation using visual odometry," in *American Controls Conference*, 2007.

[11] S. J. Julier and J. K. Uhlmann, "A new extension of the kalman filter to nonlinear systems," in *The Proceedings of AeroSense: The 11th International Symposium on Aerospace/Defense Sensing, Simulation and Controls, Multi Sensor Fusion, Tracking and Resource Management II*, 1997.

[12] G. Hu, W. Dixon, S. Gupta, and N. Fitz-Coy, "A quaternion formulation for homography-based visual servo control," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, May 15-19, 2006, pp. 2391–2396.

[13] S. Mehta, K. Kaiser, N. Gans, and D. W.E., "Homography-based coordinate relationships for unmanned air vehicle regulation." in *AIAA Guidance, Navigation, and Control Conference and Exhibit.* AIAA, August 2006.

[14] S. Mehta, W. Dixon, D. Mac Arthur, and C. Crane, "Visual servo control of an unmanned ground vehicle via a moving airborne monocular camera," in *American Control Conference, 2006*, 14-16 June 2006, p. 6pp.

[15] K. Kaiser, N. Gans, and W. Dixon, "Position and orientation of an aerial vehicle through chained, vision-based pose reconstruction," in *AIAA GNC Conference and Exhibit.* AIAA, August 2006.

[16] V. Chitrakaran, D. Dawson, J. Chen, and W. Dixon, "Euclidean position estimation of features on a moving object using a single camera: a lyapunov-based approach," in *American Control Conference, 2005. Proceedings of the 2005*, 8-10 June 2005, pp. 4601–4606vol.7.

[17] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[18] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the 4th Alvey Vision Conference*, 1988, pp. 147–151.

[19] J. Bouguet *et al.*, "Pyramidal implementation of the lucas kanade feature tracker description of the algorithm," *Intel Corporation, Microprocessor Research Labs*, 2000.

[20] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[21] D. B. Barber, J. D. Redding, T. W. McLain, R. W. Beard, and C. N. Taylor, "Vison-based target geo-location using a fixed-wing miniature air vehicle." *Journal of Intelligent and Robotic Systems*, vol. 47, no. 3, pp. 361–382, December 2006.

[22] J. B. Saunders, B. Call, A. Curtis, R. W. Beard, and T. W. McLain, "Static and dynamic obstacle avoidance in miniature air vehicles," in *AIAA 5th Aviation, Technology, Integration, and Operations Conference*, Sep 2005.

[23] R. S. Christiansen, "Design of an Autopilot for Small Unmanned Aerial Vehicles," Master's thesis, Brigham Young University, August 2004.

[24] J. Volpe, "Vulnerability assessment of the transport infrastructure relying on the global positioning system," Office of the Assistant Secretary for Transportation Policy, U.S. Department of Transportation, Center, J. A. V. N. T. S., Tech. Rep., Aug 2001.

[25] J.-H. Kim and S. Sukkarieh, "Airborne Simultaneous Localisation and Map Building," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 1, 2003, pp. 406–411 vol.1.

[26] M. Bryson and S. Sukkarieh, "Bearing-Only SLAM for an Airborne Vehicle," in *Australasian Conference on Robotics and Automation*, 2005.

[27] J. Kim and S. Sukkarieh, "SLAM aided GPS/INS Navigation in GPS Denied and Unknown Environments," in *The 2004 International Symposium on GNSS/GPS*, 2004.

[28] J. Langelaan and S. Rock, "Passive gps-free navigation for small uavs," in *Proc. IEEE Aerospace Conference*, 2005, pp. 1–9.

[29] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, June 2007.

[30] F. Dellaert, S. Thrun, and C. Thorpe, "Jacobian Images of Super-Resolved Texture Maps for Model-Based Motion Estimation and Tracking," in *1998 IEEE Workshop on Applications of Computer Vision*. Princeton, NJ: IEEE Computer Society, October 1998, pp. 2–7.

[31] F. Dellaert, C. Thorpe, and S. Thrun, "Super-Resolved Texture Tracking of Planar Surface Patches," in *1998 IEEE/RSJ International Conference on Intelligent Robotic Systems*, vol. 1, October 1998, pp. 197–203.

[32] P. Corke, J. Lobo, and J. Dias, "An Introduction to Inertial and Visual Sensing," *The International Journal of Robotics Research*, vol. 26, no. 6, pp. 519–535, 2007.

[33] S. Roumeliotis, A. Johnson, and J. Montgomery, "Augmenting inertial navigation with image-based motion estimation," in *2002 IEEE International Conference on Robotics and Automation*, vol. 4, 2002, pp. 4326–4333.

[34] D. D. Diel, "Stochastic Constraints for Vision Aided Inertial Navigation," Master's thesis, MIT, Jan 2005.

[35] D. S. Bayard and P. B.Brugarolas, "An Estimation Algorithm for Vision-Based exploration of small bodies in space," in *2005 American Control Conference*, 2005.

[36] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided intertial navigation," in *2007 IEEE International Conference on Robotics and Automation*, 2007.

[37] L. Armesto, J. Tornero, and M. Vincze, "Fast Ego-motion Estimation with Multi-rate Fusion of Inertial and Vision," *The International Journal of Robotics Research*, vol. 26, no. 6, pp. 577–589, 2007. [Online]. Available: http://ijr.sagepub.com/cgi/content/abstract/26/6/577

[38] T. Viéville, E. Clergue, and P. Facao, "Computation of ego-motion and structure from visual and inertial sensors using the vertical cue," in *International Conference on Computer Vision*, 1993, pp. 591–598.

[39] J. Domke and Y. Aloimonos, "Integration of Visual and Inertial Information for Egomotion: A Stochastic Approach," in *2006 IEEE International Conference on Robotics and Automation*, 2006, pp. 2053–2059.

[40] J. Lobo and J. Dias, "Vision and inertial sensor cooperation using gravity as a vertical reference," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 12, pp. 1597–1608, 2003.

[41] ——, "Inertial sensed ego-motion for 3d vision," *Journal of Robotic Systems*, vol. 21, no. 1, pp. 3–12, 2004.

[42] ——, "Relative Pose Calibration Between Visual and Inertial Sensors," *The International Journal of Robotics Research*, vol. 26, no. 6, pp. 561–575, 2007. [Online]. Available: http://ijr.sagepub.com/cgi/content/abstract/26/6/561

[43] D. Nister, O. Naroditsky, and J. Bergen, "Visual Odometry," in *2004 IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2004.

[44] ——, "Visual odometry for ground vehicle applications," *Journal of Field Robotics*, vol. 23, no. 1, pp. 3–20, Jan 2006.

[45] T. Kanade, O. Amidi, and Q. Ke, "Real-time and 3D Vision for Autonomous Small and Micro Air Vehicles," in *2004 IEEE Conference on Decision and Control*, vol. 2, 2004, pp. 1655–1662.

[46] J. Kehoe, R. Causey, A. Arvai, and R. Lind, "Partial Aircraft State Estimation from Optical Flow Using Non-Model-Based Optimization," in *2006 American Control Conference*, 2006, p. 6.

[47] J. Kehoe, A. Watkins, R. Causey, and R. Lind, "State Estimation using Optical Flow from Parallax-Weighted Feature Tracking," in *2006 AIAA Guidance, Navigation, and Control Conference*, 2006.

[48] K. Kaiser, N. Gans, and W. Dixon, "Position and orientation of an aerial vehicle through chained, vision-based pose reconstruction," in *2006 AIAA Guidance, Navigation, and Control Conference*. AIAA, Aug 2006.

[49] ——, "Localization and Control of an Aerial Vehicle through Chained, Vision-Based Pose Reconstruction," in *2007 American Control Conference*, 2007, pp. 5934–5939.

[50] D. Nister, "An Efficient Solution To The Five-Point Relative Pose Problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 756–770, 2004.

[51] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry, *An Invitation to 3-D Vision: From Images to Geometric Models*. SpringerVerlag, 2004.

[52] R. Eustice, H. Singh, J. Leonard, M. Walter, and R. Ballard, "Visually navigating the rms titanic with slam information filters," in *Robotics: Science and Systems*, Cambridge, Mass, Jun 2005.

[53] K. Richmond and S. Rock, "A real-time visual mosaicking and navigation system," *Unmanned Untethered Submersible Technology*, 2005.

[54] S. Fleischer, "Bounded-Error Vision-Based Navigation Of Autonomous Underwater Vehicles," Ph.D. dissertation, stanford university, 2000.

[55] S. Baker and I. Matthews, "Lucas-Kanade 20 years on: A Unifying Framework. Part 1: The Quantity Approximated, the Warp Update Rule, and the Gradient Descent Approximation," *International Journal of Computer Vision*, vol. 56, no. 3, pp. 221–255, February 2004.

[56] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[57] J. Kim and S. Sukkarieh, "Robust Multi-loop Airborne SLAM in Unknown Wind Environments," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 2006, pp. 1536–1541.

[58] D. Strelow, "Motion estimation from image and inertial measurements," Ph.D. dissertation, Carnegie Mellon University, 2004. [Online]. Available: http://www.dennis-strelow.com/publications/index.html

[59] M. J. Veth, J. F. Raquet, and M. Pachter, "Stochastic constraints for efficient image correspondence search," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, pp. 973–982, 2006.

[60] M. Veth and J. Raquet, "Fusing low-cost image and inertial sensors for passive navigation," *Journal of the Institute of Navigation*, vol. 54, no. 1, pp. 11–20, 2007.

[61] J. Langelaan, "State estimation for autonomous flight in cluttered environments," Ph.D. dissertation, Stanford University, 2006.

[62] R. Prazenica, A. Watkins, A. Kurdila, Q. Ke, and T. Kanade, "Vision-based kalman filtering for aircraft state estimation and structure from motion," in *2005 AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2005, pp. 1–13.

[63] S. Soatto, R. Frezza, and P. Perona, "Motion estimation via dynamic vision," *IEEE Transactions on Automatic Control*, vol. 41, no. 3, pp. 393–413, 1996.

[64] B. B. Ready and C. N. Taylor, "Improving accuracy of MAV pose estimation using visual odometry," in *American Control Conference, 2007. ACC '07*, 2007, pp. 3721–3726.

[65] E. D. Andersen and C. N. Taylor, "Improving MAV pose estimation using visual information," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[66] H. W. Sorenson, "Kalman filtering techniques," in *Advances in Control Systems, Theory and Applications*, C. T. Leondes, Ed., vol. 3, 1966, pp. 218–292.

[67] Z. Xing and D. Gebre-Egziabher, "Modeling and bounding low cost inertial sensor errors," in *Proc. IEEE/ION Position, Location and Navigation Symposium*, 2008, pp. 1122–1132.

[68] "Ieee standard specification format guide and test procedures for single axis interferometric fiber optic gyros. *i*eee standard 952-1997."

[69] D. H. Titterton and J. L. Weston, *Strapdown Inertial Navigation Technology*. Peter Peregrinus Ltd., 1997.

[70] M. Goodrich, B. Morse, D. Gerhardt, J. Cooper, M. Quigley, J. Adams, and C. Humphrey, "Supporting wilderness search and rescue using a camera-equipped mini UAV: Research Articles," *Journal of Field Robotics*, vol. 25, no. 1&dash, pp. 89–110, 2008.

[71] J. Bradley and C. Taylor, "Particle filter based mosaicking for tracking forest fires," in *AIAA Guidance, Navigation, and Control Conference*, 2007.

[72] S. Herwitz, L. Johnson, J. Arvesen, R. Higgins, J. Leung, and S. Dunagan, "Precision Agriculture as a Commercial Application for Solar-Powered Unmanned Aerial Vehicles," *AIAA 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles*, 2002.

[73] M. Veth and J. Raquet, "Two-dimensional stochastic projections for tight integration of optical and inertial sensors for navigation," in *National Technical Meeting Proceedings of the Institute of Navigation*. Air Force Institute of Technology, 2006, DTIC Research Report, pp. 587–596.

[74] D. Strelow and S. Singh, "Long-term motion estimation from images," in *In Proceedings, International Symposium on Experimental Robotics*, July 2006.

[75] A. Brown, B. Bockius, B. Johnson, H. Holland, and D. Wetlesen, "Flight test results of a video-aided gps/inertial navigation system," in *Proceedings of the 2007 ION GNSS Conference*, 2007, pp. 1111–1117.

[76] G. Rees and R. Alexander, "A framework for assessing and designing vision-based slam systems for autonomous vehicles," in *Proceedings, SEAS-DTC Technical Conference*, 2007. [Online]. Available: http://www.seasdtc.com/events/2007_conference/papers/AA004.pdf

[77] S. Huang and G. Dissanayake, "Convergence and Consistency Analysis for Extended Kalman Filter Based SLAM," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 1036–1049, 2007.

[78] P. Gibbens, G. Dissanayake, and H. Durrant-Whyte, "A closed form solution to the single degree of freedom simultaneouslocalisation and map building (SLAM) problem," in *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, vol. 1, 2000.

[79] P. Maybeck, *Stochastic models, estimation and control. Vol. 1*. Academic Press London, 1982.

[80] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems," in *Proc. SPIE Vol. 3068, p. 182-193, Signal Processing, Sensor Fusion, and Target Recognition VI, Ivan Kadar; Ed.*, ser. Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference, I. Kadar, Ed., vol. 3068, Jul. 1997, pp. 182–193.

[81] A. Doucet and N. De Freitas, *Sequential Monte Carlo Methods in Practice*. Springer, 2001.

[82] S. Julier and J. LaViola, "On kalman filtering with nonlinear equality constraints," *IEEE Journal of Signal Processing*, vol. 55, no. 6, pp. 2774–2784, 2007.

[83] H. Durrant-Whyte, D. Rye, and E. Nebot, "Localization of Autonomous Guided Vehicles," *Proceedings of the 8th International Symposium on Robotics Research (G. Hirzinger and G. Giralt, eds.), Springer Verlag, New York*, pp. 613–625, 1995.

[84] P. Piniés, T. Lupton, S. Sukkarieh, and J. D. Tardós, "Inertial aiding of inverse depth slam using a monocular camera," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 2797–2802.

[85] J. M. M. Montiel, J. Civera, and J. Davison, "Unified inverse depth parametrization for monocular slam," *Robotics Science and Systems*, vol. 9, p. 1, 2006.

[86] D. M. Bevly, D. Gebre-Egziabher, and B. Parkinson, "Parametric error equations for dead reckoning navigators used in ground vehicle guidance and control," *Navigation*, vol. 53, pp. 135–147, 2006.

[87] C. N. Taylor, "Enabling navigation of mavs through inertial, vision, and air pressure sensor fusion," *Accepted in Lecture Notes in Electrical Engineering*, vol. 35, 2009.

[88] A. D. King, "Inertial navigation-forty years of evolution," in *General Electric Company Review, vol. 13, no. 3*, 1998.

[89] R. Tranfield, "Ins/gps navigation systems for land applications," in *IEEE Position Location and Navigation Symposium, vol. 22-26, pp. 391 - 398*, 1996.

[90] M. George and S. Sukkarieh:, "Inertial navigation aided by monocular camera observations of unknown features," in *2007 IEEE International Conference on Robotics and Automation, Roma, Italy, 10-14 April*, 2007.

[91] P. D. D. Diel and S. Teller, ": Epipolar constraints for vision-aided inertial navigation," in *Proc. 7th IEEE Workshop on Applications of Computer Vision January, pp. 221-228*, 2005.

[92] D. W. D. Richards and J. Ross, "Evolving cooperative strategies for mav teams," in *GECCO*, 2005.

[93] R. Sharma and D. Ghose, "Collision avoidance between uav clusters using swarm intelligence techniques," *International Journal of Systems Science*, vol. 40,5, pp. 521–538, 2009.

[94] P. DeJong, "Coalition Formation in Multi-Agent UAV Systems," Ph.D. dissertation, University of Central Florida Orlando, Florida, 2005.

[95] T. McLain, P. Chandler, S. Rasmussen, and M. Pachter, "Cooperative control of UAV rendezvous," in *American Control Conference, 2001. Proceedings of the 2001*, vol. 3, 2001.

[96] R. Kurazume, S. Hirose, S. Nagata, and N. Sashida, "Study on cooperative positioning system (basic principle and measurement experiment)," in *Proc. IEEE International Conference on Robotics and Automation*, vol. 2, 22–28 April 1996, pp. 1421–1426.

[97] R. Kurazume and S. Hirose, "Study on cooperative positioning system: optimum moving strategies for cps-iii," in *Proc. IEEE International Conference on Robotics and Automation*, vol. 4, 16–20 May 1998, pp. 2896–2903.

[98] I. M. Rekleitis, G. Dudek, and E. E. Milios, "Multi-robot cooperative localization: a study of trade-offs between efficiency and accuracy," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and System*, vol. 3, 30 Sept.–5 Oct. 2002, pp. 2690–2695.

[99] L. Merino, J. Wiklund, F. Caballero, A. Moe, J. De Dios, P. Forssen, K. Nordberg, and A. Ollero, "Vision-based multi-UAV position estimation," *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, pp. 53–62, 2006.

[100] S. I. Roumeliotis and G. A. Bekey, "Collective localization: a distributed kalman filter approach to localization of groups of mobile robots," in *Proc. IEEE International Conference on Robotics and Automation ICRA '00*, vol. 3, 24–28 April 2000, pp. 2958–2965.

[101] G.Wei, "Adaption and learning in multi-agent systems: Some remarks and a bibliography," *Lecture Notes in Computer Science 1042 (Springer, Berlin)*, pp. 1–21, 1996.

[102] R. Kurazume, S. Nagata, and S. Hirose, "Cooperative positioning with multiple robots," in *Proc. IEEE International Conference on Robotics and Automation*, 8–13 May 1994, pp. 1250–1257.

[103] A. C. Sanderson, "A distributed algorithm for cooperative navigation among multiple mobile robots," *Advanced Robotics*, vol. 12, pp. 335–349, 1998.

[104] J. Spletzer, A. Das, R. Fierro, C. Taylor, V. Kumar, and J. Ostrowski, "Cooperative localization and control for multi-robot manipulation," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, 2001, pp. 631–636 vol.2.

[105] P. Nebot, D. Gomez, and E. Cervera, "Agents for cooperative heterogeneous mobile robotics: a case study," in *Proc. IEEE International Conference on Systems, Man and Cybernetics*, vol. 1, 5–8 Oct. 2003, pp. 557–562.

[106] A. Bahr, J. J. Leonard, and M. F. Fallon, "Cooperative localization for autonomous underwater vehicles," *International Journal of Robotics Research*, vol. Volume 28 , Issue 6, pp. 714–728, 2009.

[107] R. Sharma and C. Taylor, "Cooperative navigation of mavs in gps denied areas," in *Proc. IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems MFI 2008*, 20–22 Aug. 2008, pp. 481–486.

[108] A. Mourikis and S. Roumeliotis, "Analysis of positioning uncertainty in reconfigurable networks of heterogeneous mobile robots," in *Proc. IEEE International Conference on Robotics and Automation ICRA '04*, vol. 1, 2004, pp. 572–579 Vol.1.

[109] ——, "Performance analysis of multirobot cooperative localization," *IEEE Transaction for Robotics and Autonomus Systems*, vol. 22, no. 4, pp. 666–681, 2006.

[110] S. Y. Cho, B. D. Kim, Y. S. Cho, and W. S. Choi, "Observability analysis of the ins/gps navigation system on the measurements in land vehicle applications," in *Proc. International Conference on Control, Automation and Systems ICCAS '07*, 17–20 Oct. 2007, pp. 841–846.

[111] F. M. Mirzaei and S. I. Roumeliotis, "A kalman filter-based algorithm for imu-camera calibration: Observability analysis and performance evaluation," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1143–1156, Oct. 2008.

[112] M. Bryson and S. Sukkarieh, "Observability analysis and active control for airborne slam," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, no. 1, pp. 261–280, January 2008.

[113] T. Vidal-Calleja, M. Bryson, S. Sukkarieh, A. Sanfeliu, and J. Andrade-Cetto, "On the observability of bearing-only slam," in *Proc. IEEE International Conference on Robotics and Automation*, 10–14 April 2007, pp. 4114–4119.

[114] X. S. Zhou and S. I. Roumeliotis, "Robot-to-robot relative pose estimation from range measurements," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1379–1393, Dec. 2008.

[115] R. Sharma and C. N. Taylor, "Vision-based distributed cooperative navigation for mavs in gps denied areas," in *Infotech@Aerospace*, 2009.

[116] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation*, T. Kailath, Ed. Prentice Hal, 2001.

[117] R. Hermann and A. Krener, "Nonlinear controllability and observability," *IEEE Transactions on Automatic Control*, vol. 22, no. 5, pp. 728–740, Oct 1977.

Appendix A

In Section VI-A2, we made the claim that $E[\cos(\mathcal{N}(0,\sigma^2))] = e^{-\sigma^2/2}$. This result was determined experimentally as we explain in this paragraph. In Figure 44, we plot the expected value of taking the cosine of a Gaussian random process. The x-axis represents the standard deviation of the random process, while the plot shows the mean of ten million random numbers run through the cosine. From this plot, it was estimated that the form of the expected value is an exponential. Using an iterative least squares solution to match the expected values with an exponential curve, we found that $E[\cos(\mathcal{N}(0,\sigma^2))] = e^{-\sigma^2/2}$. We also observed that as the number of random numbers used to compute the expected value was increased, the more closely the exponential term and the expected value tracked each other, providing verification of the equality introduced above.

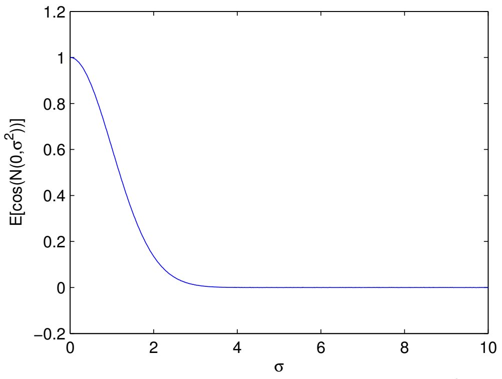


Fig. 44. The plot of the expected value of $\cos(\mathcal{N}(0,\sigma^2))$, where the x axis is the σ used to compute the expected value. This plot shows the mean across 10 million cosines of Gaussian random variables at each sigma value.